PATENT

<u>IN THE UNITED STATES PATENT AND TRADEMARK OFFICE</u>

In re application of:	)
	)
Donham et al.	) Group Art Unit: 2628
	)
Application No.: 10/006,551	) Examiner: Amin, Jwalant B.
	)
Filed: 11/30/2001	) Atty Docket No.:
	) NVIDP064/P000286
For: SYSTEM, METHOD AND COMPUTER	)
PROGRAM PRODUCT FOR USING	) Date: 08/26/2008
TEXTURES AS INSTRUCTIONS FOR	)
GRAPHICS PROCESSING	)
	)

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

ATTENTION: Board of Patent Appeals and Interferences

REPLY BRIEF (37 C.F.R. § 41.37)

This Reply Brief is being filed within two (2) months of the mailing of the Examiner's Answer mailed on 06/26/2008.

Following is an issue-by-issue reply to the Examiner's Answer.

In the Examiner's Answer mailed 06/26/2008, the Examiner has responded to appellant's arguments within each separate group multiple times using the same and/or substantially similar arguments. In response, appellant has addressed each of the Examiner's distinct responses within each group and has shown how the prior art references are still deficient in making a prior art showing of appellant's claimed techniques.

Issue # 1:

The Examiner has rejected Claims 1-12, 18-21, 24-28, and 30 under 35 U.S.C. 103(a) as being unpatentable over Rivard (U.S. Patent No. 5,987,567), in view of Wang (U.S. Patent No. 5,831,640).

Group #1: Claims 1-12, 21, 24, and 27

To establish a *prima facie* case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art and not based on appellant's disclosure. *In re Vaeck,* 947 F.2d 488, 20 USPQ2d 1438 (Fed.Cir.1991).

With respect to the first element of the *prima facie* case of obviousness, the Examiner has argued that "[o]ne of ordinary skill in [the] art... would have expected [appellant's] invention to perform equally well with Rivard's reference that teaches to send and receive information/instruction from the texture mapping stage and texel cache system to the DRAM because using [these] components together will also result in sending and receiving instructions to and from DRAM." To the contrary, appellant respectfully asserts that it would not have been obvious to modify the teaching of Rivard, especially in view of the vast evidence to the contrary.

Appellant respectfully points out that Rivard teaches that "[b]ecause memory request generator 1020 is between cache tag blocks 1010, 1015 and cache data store 1030, generator 1020 can perform DRAM 655 memory requests before the address and instruction information reach cache data store and memory data resolver 1030" (Col. 6, lines 62-66 - emphasis added). In addition, Rivard teaches that "[o]nce memory requests are generated, there is, depending on the DRAM design, a constant latency of about five to ten clock cycles (or possibly more) which includes time for exiting and reentering the graphics pipeline hardware, to effect a page hit," and "[t]herefore, graphics accelerator system 635 includes pipeline latency elements 1025 to coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030" (Col. 6, line 66-Col. 7, line 7 - emphasis added). Furthermore, Rivard discloses that "the memory request generator is coupled between the cache tag block and the cache data store for performing a memory request before an address and instruction information associated with the needed texels reach the cache data store" (Col. 10, lines 48-52 - emphasis added).

Appellant respectfully asserts that a combination of data and instructions in Rivard would result in no need for Rivard's purposefully included "pipeline latency elements," since the combination of data and instructions would arrive together. As a result, there is no suggestion that the modification of the Rivard reference, as argued by the Examiner, is desired. The mere fact that references can be combined or modified does not render the resultant combination obvious unless the prior art also suggests the desirability of the combination. In re Mills, 916 F.2d 680, 16 USPQ2d 1430 (Fed. Cir. 1990). Although a prior art device "may be capable of being modified to run the way the apparatus is claimed, there must be a suggestion or motivation in the reference to do so." 916 F.2d at 682, 16 USPQ2d at 1432.).

Additionally, appellant respectfully notes that modifying the Rivard reference, as suggested by the Examiner, would change the principle of operation of the Rivard system, since it would obviate the need for the purposefully included "pipeline latency elements 1025," as noted above. If the proposed modification or combination of the prior art would change the principle of operation of the prior art invention being modified, then the teachings of the references are not sufficient to render the claims prima facie obvious. In re Ratti, 270 F.2d 810, 123 USPQ 349 (CCPA 1959)

In the Office Action mailed 09/20/2007, the Examiner has failed to specifically respond to appellant's above arguments. Again, appellant respectfully asserts that modifying the Rivard reference, as suggested by the Examiner, would change the principle of operation of the Rivard system, since it would obviate the need for the purposefully included "pipeline latency elements 1025," as noted above.

In the Examiner's Answer mailed 06/26/2008, on Pages 11-13, the Examiner has argued the following:

> "However, the examiner interprets that Rivard does not explicitly teach texture mapping stage and texel cache system together functions as a texture module. However, it would have been obvious to one of ordinary skill in art at the time of present invention to combine texture mapping stage and texel cache system of Rivard to work together as a texture module (it should be noted that the examiner interprets texture mapping stage and texel cache system combined together will perform functions similar to the claimed texture module, and therefore it would be obvious to modify the Rivard reference). The unity of diversity of parts would depend more upon the choice of the manufacturer, and the convenience and availability of the machines and tools necessary to construct the texture module, than on any inventive concept. One of ordinary skill in art, furthermore, would have expected [appellant's] invention to perform equally well with Rivard's reference that teaches to send and receive information/instruction from the texture mapping stage and texel cache system to the DRAM because using this components together will also result in sending and receiving instructions to and from DRAM. Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to modify Rivard to obtain the invention as specified in the claim.
> Rivard also teaches the arrival of memory data and associated instructions at the cache data store and memory data resolver component of the text cache system is coordinated by pipeline latency elements (col. 7 lines 4-7; memory data has associated instructions; it should be noted that depending on the DRAM design, there could be a constant latency to effect a page hit; however, in the instance when the instructions and data are combined together, no pipeline latency element will be required; therefore, when the Rivard reference is modified as taught by Wang below, the modified system might have the data and the instructions coordinated when arrived at cache data store and memory data resolver, and will not require the use of pipeline latency element; however, having the pipeline latency element will help to coordinate the instructions and data at their arrival at cache data store and memory data resolver, when there is a delay between the arrival of data and it's associated instructions; it should be noted that Rivard has this pipeline latency element in addition to the claimed limitation, and moreover, the examiner interprets that the modified system with pipeline latency element will still function as effectively and generate the desired results, even when there is no delay between the arrival of the data and it's associated instructions). Although Rivard discloses the limitations as stated above, Rivard does not explicitly teach that the memory returns

instructions along with data, in response to instruction request from the texture module. However, Wang teaches a graphics subunit (texture module) in a graphics hardware system that supplies data and control signals to local frame buffer memory for executing a series of display instructions (Fig. 1, col. 5 lines 38-67; the computer memory includes the local frame buffer memory, which corresponds to the DRAM; based on the control signals send by the graphics hardware system, the frame buffer returns the polygon display instructions that includes the texture data, so that the graphics subunit executes the display instructions using this data; it should be noted that there is nothing in the teaching of either Rivard or Wang that teaches against such a modification of Rivard's system). Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device (col. 5 lines 43-45 and lines 63-67)."

Appellant respectfully disagrees and points out that Rivard teaches that "[b]ecause memory request generator 1020 is between cache tag blocks 1010, 1015 and cache data store 1030, generator 1020 can perform DRAM 655 memory requests before the address and instruction information reach cache data store and memory data resolver 1030" (Col. 6, lines 62-66 - emphasis added). In addition, Rivard teaches that "accelerator system 635 includes pipeline latency elements 1025 to coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030" (Col. 6, line 66-Col. 7, line 7 - emphasis added). Additionally, Wang teaches "a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" (Col. 5, lines 40-46 - emphasis added) and that "[m]any of the polygon display instructions include texture data to be displayed within the polygon" (Col. 5, lines 40-48 - emphasis added).

However, appellant asserts that the combination of Rivard and Wang, as suggested by the Examiner, would obviate the need for Rivard's pipeline latency elements 1025 that coordinate arrival of the memory data and of the associated instructions, since Wang's executed display instructions may include texture data and thus would not need to be coordinated in pipeline latency elements 1025, as in Rivard. Therefore, it would not have "been obvious to one of ordinary skill in art… to have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device," as alleged by the Examiner. Again, appellant asserts that executing Wang's display instructions that may include texture data would clearly obviate the need for Rivard's purposefully included "pipeline latency elements 1025," as noted above. If the proposed modification or combination of the prior art would change the principle of operation of

the prior art invention being modified, then the teachings of the references are not sufficient to render the claims *prima facie* obvious. *In re Ratti*, 270 F.2d 810, 123 USPQ 349 (CCPA 1959)

Furthermore, appellant respectfully asserts that, in the Abstract, Rivard teaches "[a] system for caching texel information in a cache data store, for use in a graphics rendering system which uses interpolative sampling to compute texture color values." Additionally, Rivard teaches that "[t]he system includes a texel memory storing texel information, a graphics application program for using interpolative sampling to compute dynamic texture values, a first cache data storage for a number of the most-recently-retrieved texels, a second cache data storage for a previously-retrieved adjacent line of texels, cache tag blocks for determining whether the texels needed by the graphics accelerator system are cached in either of the first or second cache data stores, and a memory request generator for retrieving texels from texel memory upon indication of a miss by the cache tag blocks."

However, Rivard does not recognize one of the various possible problems solved by appellant, namely to "accommodate the programmability of recent texture and shader modules without being inhibited by the size of associated programs," for example. It was this insight in solving this problem that helped the inventors conceive of the claimed invention which overcomes the drawbacks of the prior art. "Because that insight was contrary to the understandings and expectations of the art, the structure effectuating it would not have been obvious to those skilled in the art." *Schenck v. Nortron Corp.*, 713 F.2d at 785, 218 USPQ at 700 (citations omitted).

In the Office Action mailed 09/20/2007, in paragraph 3 on Page 2, the Examiner has argued the following:

> "In response to [appellant's] argument that the references fail to show certain features of [appellant's] claimed invention, it is noted that the features upon which [appellant] relies (i.e., to accommodate the programmability of recent texture and shader modules without being inhibited by the size of the associated programs) are not recited in the rejected claims(s). Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993)."

Appellant respectfully asserts that Rivard teaches "[a] system for caching texel information in a cache data store, for use in a graphics rendering system which uses interpolative sampling to

compute texture color values" (Abstract). Clearly, Rivard's system for caching texel information in a cache data store does not recognize one of the various possible problems solved by appellant, namely to "accommodate the programmability of recent texture and shader modules without being inhibited by the size of associated programs," as disclosed in the Specification, Page 5, lines 13-15, for example. It was the <u>insight</u> in solving the aforementioned problem that helped the inventors <u>conceive</u> of the <u>claimed invention</u>, which overcomes the drawbacks of the prior art. "Because that insight was contrary to the understandings and expectations of the art, the structure effectuating it would not have been obvious to those skilled in the art." *Schenck v. Nortron Corp.*, 713 F.2d at 785, 218 USPQ at 700 (citations omitted).

In the Examiner's Answer mailed 06/26/2008, on Page 14, the Examiner has argued the following:

> "In response to [appellant's] argument that the references fail to show certain features of [appellant's] invention, it is noted that the features upon which [appellant] relies (i.e., to accommodate the programmability of recent texture and shader modules without being inhibited by the size of associated programs) are not recited in the rejected claim(s). Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir.1993). Moreover, it should be noted that Rivard does not need to teach any limitations that is not claimed by appellant, and also Rivard does not need to solve similar problems as solved by appellant's claimed invention."

Appellant respectfully disagrees and asserts that appellant is stating that the problem of "accommodat[ing] the programmability of recent texture and shader modules without being inhibited by the size of associated programs," as disclosed in the Specification, Page 5, lines 13-15, for example, is clearly not recognized by Rivard, which simply teaches a "[a] system for <u>caching texel information in a cache data store</u>, for use in a graphics rendering system which uses interpolative sampling to compute texture color values" (Abstract). Again, appellant asserts that it was the <u>insight</u> in solving the aforementioned problem that helped the inventors <u>conceive</u> of the <u>claimed invention</u>, which overcomes the drawbacks of the prior art. "Because that insight was contrary to the understandings and expectations of the art, the structure effectuating it would not have been obvious to those skilled in the art." *Schenck v. Nortron Corp.*, 713 F.2d at 785, 218 USPQ at 700 (citations omitted).

Additionally, appellant respectfully asserts that the following excerpts from Rivard demonstrate that such reference, in fact, *teaches away* from appellant's claimed invention.

"If the interface to DRAM 655 is 32-bits wide, the texture mode indicates a 16-bit per pixel texture lookup and the data is conveniently aligned in the texture map, then it is possible to satisfy two lookup requests with a single read request. However, if the data is not aligned, then two read requests are needed." (Col. 6, lines 56-61 - emphasis added)

"Therefore, graphics accelerator system 635 includes pipeline latency elements 1025 to coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030." (Col. 7, lines 3-7 - emphasis added)

"...the memory request generator is coupled between the cache tag block and the cache data store for performing a memory request before an address and instruction information associated with the needed texels reach the cache data store." (Col. 10, lines 48-52 - emphasis added)

Appellant respectfully asserts that the "pipeline latency elements 1025," as described in Rivard, are introduced specifically "to coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030" (emphasis added). Thus, Rivard actually *teaches away* from appellant's claim language by intentionally incorporating the "pipeline latency elements 1025" for the specific purpose of coordinating the arrival of the instructions and the memory data from separate sources. A *prima facie* case of obviousness may also be rebutted by showing that the art, in any material respect, *teaches away* from the claimed invention. *In re Geisler*, 116 F.3d 1465, 1471, 43 USPQ2d 1362, 1366 (Fed. Cir. 1997).

In the Office Action mailed 09/20/2007, the Examiner has failed to specifically respond to appellant's above arguments. Again, appellant respectfully asserts that Rivard actually *teaches away* from appellant's claim language by intentionally incorporating the "pipeline latency elements 1025" for the specific purpose of coordinating the arrival of the instructions and the memory data from separate sources.

In the Examiner's Answer mailed 06/26/2008, the Examiner has again has failed to specifically respond to appellant's above arguments. Again, appellant respectfully asserts that Rivard actually *teaches away* from appellant's claim language by intentionally incorporating the

"pipeline latency elements 1025" for the specific purpose of coordinating the arrival of the instructions and the memory data from separate sources.

Thus, clearly at least the first element of the *prima facie* case of obviousness has not been met by the Rivard reference.

More importantly, with respect to the third element of the *prima facie* case of obviousness, appellant respectfully asserts that the Rivard reference also fails to meet all of appellant's claim limitations. Specifically, with respect to independent Claims 1, 24, and 27, the Examiner has relied on Figure 6 and Figure 10, in addition to Col. 4, lines 46-57; and Col. 6, line 45-Col. 7, line 10 (excerpted below) from Rivard to make a prior art showing of appellant's claimed "sending an instruction request to video memory, where a texture module... sends the instruction request to the video memory" (see this or similar, but not necessarily identical language in the foregoing independent claims).

"Graphics application program 670 transfers graphical information from data storage 625 into DRAM 655 for local storage. Graphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650. Texture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645." (Col. 4, lines 46-57 - emphasis added)

"Cache data store 1030 responsively outputs on lines 649 the texture values cached in the read location.

If a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020. Memory request generator 1020 generates memory requests for all misses (up to four for bi-linear sampling and up to eight for tri-linear sampling), and forwards the requests to DRAM 655 for information retrieval. DRAM 655 returns the memory data on bus 660 to cache data store and memory data resolver 1030, which stores the memory data at the write address. If the interface to DRAM 655 is 32-bits wide, the texture mode indicates a 16-bit per pixel texture lookup and the data is conveniently aligned in the texture map, then it is possible to satisfy two lookup requests with a single read request. However, if the data is not aligned, then two read requests are needed.

Because memory request generator 1020 is between cache tag blocks 1010, 1015 and cache data store 1030, generator 1020 can perform DRAM 655 memory requests before the address and instruction information reach cache data store and memory data resolver 1030. Once memory requests

are generated, there is, depending on the DRAM design, a constant latency of about five to ten clock cycles (or possibly more) which includes time for exiting and reentering the graphics pipeline hardware, to effect a page hit. Therefore, graphics accelerator system 635 includes pipeline latency elements 1025 to coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030." (Col. 6, line 45-Col. 7, line 10 - emphasis added)

Appellant respectfully asserts that the figures relied on by the Examiner only generally illustrate a computer system and a block diagram detailing the graphics accelerator system showing the pipeline latency elements 1025. In addition, the excerpts relied on by the Examiner merely teach that the "[m]emory request generator 1020 generates memory requests for all misses (up to four for bi-linear sampling and up to eight for tri-linear sampling), and forwards the requests to DRAM 655 for information retrieval" (emphasis added). However, disclosing that a memory request is generated for all misses, as in Rivard, fails to teach "sending an instruction request to video memory, where a texture module... sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Appellant notes that the Examiner has argued that "the texture mapping stage and the texel cache system together are considered the texture module." Appellant respectfully disagrees and asserts that Rivard merely suggests that the "[g]raphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650" where the "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 49-57 - emphasis added). Clearly, the texture mapping stage is separate from the texture cache system, as clearly disclosed in Rivard, and thus fails to suggest "sending an instruction request to video memory, where a texture module... sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

In addition, the Examiner has argued that "memory requests/read requests for all misses that are forwarded to DRAM are instruction requests based on which DRAM sends back information... [where] the DRAM sends memory data based on the memory requests/read requests from the

texture module." Further, the Examiner has argued that "[m]emory requests/read requests act as an instruction to DRAM, which performs a particular function based on the request."

Appellant respectfully disagrees, and asserts that Rivard merely discloses that the "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57). Additionally, Rivard discloses that "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015" and "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" which "generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-36, and lines 48-53 – emphasis added).

However, Rivard's disclosure that the memory request generator 1020, which is included in the texel cache system (see Figure 10), generates memory requests for all misses and forwards the requests to DRAM for information retrieval, simply fails to even suggest "sending an instruction request to video memory, where a texture module... sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Moreover, the texel sample address computation block (included in texture mapping block 645) which forwards sample points to the cache tag blocks 1010 and 1015 (included in texel cache system 650), as disclosed in Rivard, fails to meet "sending an instruction request to video memory, where a texture module... sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Furthermore, the memory request to the DRAM, as in Rivard, fails to suggest "an instruction request," in the manner as claimed by appellant.

In addition, appellant respectfully asserts that the following excerpts from Rivard further demonstrate that the Rivard fails to disclose appellant's claimed "sending an instruction request to video memory, where a texture module... sends the instruction request to the video memory" (see this or similar, but not necessarily identical language in the foregoing independent claims).

```
"If the interface to DRAM 655 is 32-bits wide, the texture mode
indicates a 16-bit per pixel texture lookup and the data is
conveniently aligned in the texture map, then it is possible to satisfy
two lookup requests with a single read request. However, if the data is
```

not aligned, then two read requests are needed." (Col. 6, lines 56-61 - emphasis added)

"Therefore, graphics accelerator system 635 includes <u>pipeline latency elements 1025 to coordinate arrival of the .memory data</u> and of the <u>associated instructions</u> at cache data store and memory data resolver 1030." (Col. 7, lines 3-7 - emphasis added)

"...the memory request generator is coupled between the cache tag block and the cache data store for performing a <u>memory request before an address and instruction information associated with the needed texels</u> reach the cache data store." (Col. 10, lines 48-52 - emphasis added)

First, appellant respectfully points out that Rivard merely teaches that "the memory request generator is coupled between the cache tag block and the cache data store for performing a memory request <u>before</u> an address and instruction information associated with the needed texels reach the cache data store" (emphasis added). Clearly, coordinating the arrival of "<u>memory data</u>", where "the data is conveniently aligned in the <u>texture map</u>" (emphasis added), as in Rivard (see excerpts above), fails to teach "sending an <u>instruction request</u>" (emphasis added), as claimed by appellant.

Second, appellant respectfully asserts that Rivard simply discloses that a <u>memory request</u> is performed <u>before</u> an address and instruction information associated with the needed <u>texels</u> reach the cache data store (see excerpts above). Thus, the memory data in Rivard simply relates to texel data, which clearly fails to suggest "sending an <u>instruction request</u> to video memory, where a texture module... sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Furthermore, the Examiner has argued that "the definition of instruction provided by dictionary.com, which defines instructions as a command given to a computer to carry out a particular operation and can contain data to be used in the operation; here DRAM sends memory data based on the memory requests/read requests from the texture module." Further, the Examiner has argued that "[m]emory requests/read requests act as an instruction to DRAM, which performs a particular function based on the request."

Appellant respectfully disagrees with the Examiner's arguments and asserts that the dictionary.com reference provided by the Examiner merely defines an instruction as "a command

given to a computer to carry out a particular operation." Appellant respectfully asserts that Rivard merely discloses that the "cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" which "generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-36, and lines 48-53 – emphasis added). However, the disclosure of the memory request generator generating a memory request and forwarding the request to DRAM for information retrieval, as in Rivard, simply fails to suggest "sending an instruction request to video memory, where a texture module... sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Clearly, generating and forwarding a memory request to DRAM, as in Rivard, fails to suggest "sending an instruction request," particularly where "instructions [are received]...in response to the instruction request" (emphasis added), in the context as claimed by appellant.

In the Office Action mailed 09/20/2007, on Page 2, line 19 to Page 3, line 19; and Page 4, lines 3-7, the Examiner has argued the following:

> "However, the examiner interprets that Rivard (Fig. 6, Fig. 10, col. 4 lines 46-57, col. 6 lines 45-67, col. 7 lines 1-10) teaches sending an instruction request to video memory, where a texture module in a graphics pipeline sends the instruction request to the video memory (a texture mapping stage sends information to texel cache system; the texture mapping stage and the texel cache system together are considered as texture module, so the information is passed between the components of the texture module; texel cache system comprising the memory request generator, pipeline latency elements, and cache data store and memory data resolver forwards the memory requests/read requests to DRAM for information retrieval. [T]he examiner interprets that Rivard discloses sending an instruction request to video memory by generating memory request for all misses, and forwards the requests to DRAM for information retrieval (Fig. 6, Fig. 10, col. 4 lines 46-57, col. 6 lines 45-67, col. 7 lines 1-10). It is interpreted that an instruction request could be considered as merely a request because it is basically a request that requires DRAM to perform necessary operation. The examiner also interprets that instructions stored in DRAM that are requested are basically stored as data. It is further interpreted that memory requests/read requests act as an instruction to DRAM based on which DRAM performs a particular function. Memory requests/read requests for all misses that are forwarded to DRAM are instruction requests based on which DRAM sends back information; the examiner relies on the definition of instruction provided by dictionary.com, which defines instructions as a command given to a computer to carry out a particular operation and can contain data to be used in the operation; here DRAM sends memory based on the memory requests/read requests from the texture module." (Page 2, line 19 to Page 3, line 19)

"In response to [appellant's] arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986)." (Page 4, lines 3-7)

First, appellant notes that the Examiner's arguments on Page 4, line 8 to Page 5, line 8 in the Office Action mailed 09/20/2007 are substantially similar to the arguments made by the Examiner on Page 2, line 19 to Page 3, line 19 excerpted above. Second, appellant has clearly responded to such arguments above regarding how the excerpts repeatedly relied upon by the Examiner fail to meet or suggest appellant's claimed "sending <u>an instruction request</u> to video memory, where a texture module... sends <u>the instruction request</u> to the video memory" (emphasis added), as claimed by appellant.

Further, appellant respectfully disagrees with the Examiner's arguments and asserts that the figures and excerpts from Rivard relied upon by the Examiner merely disclose that "the texture mode indicates a <u>16-bit per pixel texture lookup</u>" in addition to "satisfy[ing] two lookup requests with a single <u>read request</u>" (Col. 6, lines 56-61 – emphasis added). Additionally, the excerpts from Rivard teach that "<u>pipeline latency</u> elements 1025... coordinate arrival of the <u>memory data</u> and of the <u>associated instructions</u>" (Col. 7, lines 3-7 – emphasis added) such that "the memory request generator... perform[s] a <u>memory request before an address and instruction information associated with the needed texels</u> reach the cache data store" (Col. 10, lines 48-52 – emphasis added). Clearly, the texture mode indicating <u>a texture lookup</u>, in addition to the memory request generator performing <u>a memory request</u>, as in Rivard, simply fails to support the Examiner's allegation that "a texture module... sends the <u>instruction request</u> to the video memory" (emphasis added), and especially does not teach "sending <u>an instruction request</u> to video memory, where a texture module... sends <u>the instruction request</u> to the video memory" (emphasis added), as claimed by appellant.

In addition, Rivard's disclosure that the "<u>cache tag blocks 1010 and 1015 forward</u> a cache write address to <u>memory request generator</u> 1020," which "<u>generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval</u>" (Col. 6, lines 48-53 – emphasis added). However, the disclosure of the memory request generator generating a <u>memory request</u> and forwarding the <u>request</u> to DRAM <u>for information retrieval</u>, as in Rivard,

fails to support the Examiner's allegation "that Rivard discloses sending <u>an instruction request</u> to video memory by generating memory request for all misses, and forwards the requests to DRAM for information retrieval" (emphasis added). Clearly, generating a <u>memory request</u> and forwarding the <u>request</u> to DRAM <u>for information retrieval</u>, as in Rivard, simply fails to specifically teach "sending <u>an instruction request</u> to video memory, where a texture module... sends <u>the instruction request</u> to the video memory" (emphasis added), as claimed by appellant.

Still yet, appellant respectfully asserts that the excerpts from Rivard relied upon by the Examiner disclose that the "[g]raphics application program 670 <u>transfers graphical information</u> from data storage 625 into DRAM 655 for local storage" (Col. 4, lines 46-48 – emphasis added). Additionally, the excerpts disclose that "[i]f a hit occurs, then cache tag blocks 1010 and 1015 forward a cache read address through memory request generator 1020 and through pipeline latency elements 1025 to cache data store 1030" such that "[c]ache data store 1030 responsively outputs on lines 649 the <u>texture values cached</u> in the read location" (Col. 6, lines 42-47 – emphasis added). Clearly, Rivard is disclosing the transfer of <u>graphical information</u> into DRAM and outputting <u>cached texture values</u>, which simply fails to support the Examiner's allegation that "<u>instructions</u> stored in DRAM that are requested are basically stored as data" (emphasis added). Therefore, the disclosure of transferring <u>graphical information</u> into DRAM and outputting cached <u>texture values</u>, as in Rivard, simply fails to suggest "sending <u>an instruction request</u> to video memory, where a texture module... sends <u>the instruction request</u> to the video memory" (emphasis added), as claimed by appellant.

Furthermore, appellant again asserts that the dictionary.com reference provided by the Examiner merely defines an instruction as "a command given to a computer to carry out a particular operation." In addition, the excerpts from Rivard disclose that the "<u>memory request generator 1020 generates memory requests</u> for all misses... <u>and forwards the requests to DRAM</u> 655 for <u>information retrieval</u>" (Col. 6, lines 50-53 – emphasis added). However, generating and forwarding <u>memory requests</u> to DRAM for <u>information retrieval</u>, as in Rivard, fails to support the Examiner's allegation that "[m]emory requests/read requests for all misses that are forwarded to DRAM are <u>instruction requests based on which DRAM sends back information</u>" (emphasis added). Clearly, generating and forwarding <u>memory requests</u> to DRAM for <u>information retrieval</u>, as in Rivard, simply fails to specifically teach "sending <u>an instruction request</u> to video

memory, where a texture module... sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Appellant emphasizes that the memory request of Rivard fails to meet appellant's claimed "instruction request," as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 15-16, 16-18, 19-20, 20-21, 23-24 and 27-28, the Examiner has argued the following:

> "However, the examiner interprets that Rivard (Fig. 6, Fig. 10, col. 4 lines 46-57, col. 6 lines 45-67, col. 7 lines 1-10) teaches sending an instruction request to video memory, where a texture module in a graphics pipeline sends the instruction request to the video memory (a texture mapping stage sends information to texel cache system; the texture mapping stage and the texel cache system together are considered as texture module, so the information is passed between the components of the texture module; texel cache system comprising the memory request generator, pipeline latency elements, and cache data store and memory data resolver forwards the memory requests/read requests to DRAM for information retrieval; the examiner interprets that Rivard discloses sending an instruction request to video memory by generating memory request for all misses, and forwards the requests to DRAM for information retrieval). It is interpreted that an instruction request could be considered as merely a request because it is basically a request that requires DRAM to perform necessary operation. The examiner also interprets that a DRAM can store instructions as data, and that the memory request retrieves such data from DRAM. It is further interpreted that when there is a miss, a memory requests/read requests is send to DRAM, which acts as an instruction to DRAM based on which DRAM performs a particular function of returning data for further processing. Therefore, it is reasonable to interpret that memory requests/read requests for all misses that are forwarded to DRAM are instruction requests based on which DRAM sends back information to the graphics pipeline for further processing (it should be noted that the examiner relies on the definition of instruction provided by dictionary.com, which defines instructions as a command given to a computer to carry out a particular operation and can contain data to be used in the operation; in this case, when the memory/read request is send to DRAM, DRAM sends memory data based on the request to the texture module of the graphics pipeline for further processing)."

Appellant respectfully disagrees and asserts that Rivard teaches that "[g]raphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650," where "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 49-57 – emphasis added).

However, a graphics accelerator system 635 including graphics pipeline stages 640 that includes a texture mapping stage 645, where the texture mapping stage 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645, as in Rivard, simply fails to support the Examiner's assertion that "the texture mapping stage and the texel cache system together are considered as texture module," as alleged by the Examiner. Furthermore, Rivard's teachings that the graphics pipeline stages 640 include a texture mapping stage 645, where the texture mapping stage 645 sends information via bus 647 to texel cache system 650, simply fails to suggest that "a texture module ... sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Clearly, teaching that the texture mapping block 645, included in the graphics pipeline stages 640, sends information via bus 647 to the separate texel cache system 650, which is **not included** in the graphics pipeline stages 640, as in Rivard, simply fails to even suggest that "a texture module... sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Additionally, appellant respectfully asserts that Rivard teaches that, as illustrated in Fig. 10, the "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015," where the "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," and that "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that "generates memory requests for all misses..., and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-53 – emphasis added).

Therefore, Rivard teaches that the texel sample address computation block 1005 of the texture mapping block 645 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015, where if the cache tag blocks determine that the requested texel values are not stored in cache data store 1030, then the cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020 that generates memory requests for all misses and forwards the requests to DRAM 655 for information retrieval. Clearly, generating a memory request when the cache tag blocks determine that the requested texel values are not stored in cache data store, and forwarding the

memory request to DRAM for information retrieval, as in Rivard, simply fails to support the Examiner's allegation that "an <u>instruction request</u> could be considered as merely a request because it is basically <u>a request that requires DRAM to perform necessary operation</u>" (emphasis added), as alleged by the Examiner, and further fails to suggest appellant's claimed "instruction request," as claimed by appellant. Clearly, generating a memory request when the <u>requested texel values</u> are <u>not</u> stored in cache data store, as in Rivard, simply fails to even suggest appellant's claimed "instruction request," as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 18, 24-25 and 28-29, the Examiner has argued the following:

> "The examiner further interprets that Rivard does not explicitly teach texture mapping stage and texel cache system together functions as a texture module. However, it would have been obvious to one of ordinary skill in art at the time of present invention to combine texture mapping stage and texel cache system of Rivard to work together as a texture module (it should be noted that the examiner interprets texture mapping stage and texel cache system combined together will perform functions similar to the claimed texture module, and therefore it would be obvious to modify the Rivard reference). The unity of diversity of parts would depend more upon the choice of the manufacturer, and the convenience and availability of the machines and tools necessary to construct the texture module, than on any inventive concept. One of ordinary skill in art, furthermore, would have expected [appellant's] invention to perform equally well with Rivard's reference that teaches to send and receive information/instruction from the texture mapping stage and texel cache system to the DRAM because using this components together will also result in sending and receiving instructions to and from DRAM. Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to modify Rivard to obtain the invention as specified in the claim."

First, appellant respectfully agrees with the Examiner's assertion that "Rivard does not explicitly teach texture mapping stage and texel cache system together functions as a texture module," since, as argued hereinabove, Rivard teaches that <u>the texture mapping stage 645, included in the graphics pipeline stages 640</u>, sends information via bus 647 to the <u>separate texel cache system 650</u>, which is <u>not included</u> in the graphics pipeline stages 640.

Second, appellant respectfully disagrees with the Examiner's allegation that "it would have been obvious to one of ordinary skill in art at the time of present invention to combine texture mapping stage and texel cache system of Rivard to work together as a texture module," as alleged by the Examiner. Appellant asserts that Rivard teaches that "[a] system for <u>caching texel</u>

information in a cache data store, for use in a graphics rendering system," where the "system includes a texel memory storing texel information, ...a first cache data storage..., a second cache data storage..., cache tag blocks..., and a memory request generator..." (Abstract – emphasis added). Further, Rivard teaches, as illustrated in Fig. 10, that the texel cache system 650 "includes cache tags 1010, cache tags 1015, a memory request generator 1020, pipeline latency elements 1025 and cache data store and memory data resolver 1030" (Col. 6, lines 22-26), and that "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57 – emphasis added).

Therefore, Rivard is clearly teaching that the use of bus 647 for sending information from the texture mapping stage 645, included in the graphics pipeline stages 640, to the separate texel cache system 650, and the use of bus 649 for sending information from the texel cache system 650 back to texture mapping stage 645, which fails to suggest that it would be obvious "to combine texture mapping stage and texel cache system of Rivard to work together as a texture module," as alleged by the Examiner, and further fails to suggest that "a texture module... sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Furthermore, appellant thus formally requests a specific showing of the subject matter in ALL of the claims in any future action. Note excerpt from MPEP below.

"If the [appellant] traverses such an [Official Notice] assertion the examiner should cite a reference in support of his or her position." See MPEP 2144.03.

Third, appellant respectfully asserts that Rivard teaches that, as illustrated in Fig. 10, the "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015," where the "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," and that "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that "generates memory requests for all misses..., and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-53 – emphasis added). Additionally, Figs. 6 and 10 clearly illustrate that the texel sample address computation block 1005 of Fig. 10 is a part of the texture mapping

stage 645 that sends information via bus 647 to the cache tag blocks 1010 and 1015 of the texel cache system 650.

Therefore, Rivard teaches that the texel sample address computation block 1005 of the texture mapping stage 645 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015 via bus 647 (as illustrated in Fig. 10), where if the cache tag blocks determine that the requested texel values are not stored in cache data store 1030, then the cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020 that generates memory requests for all misses. However, the texture mapping stage 645 forwarding sample points to the cache tag blocks of the texel cache system 650, where the cache tag blocks forward a cache write address to the memory request generator that generates the memory request, as in Rivard, fails to support that Examiner's allegation that the Rivard "reference that teaches to send and receive information/instruction from the texture mapping stage and texel cache system to the DRAM because using this components together will also result in sending and receiving instructions to and from DRAM" (emphasis added), as alleged by the Examiner, and further fails to suggest that "a texture module... sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Furthermore, Rivard teaches that "[i]f the interface to DRAM 655 is 32-bits wide, the texture mode indicates a 16-bit per pixel texture lookup and the data is conveniently aligned in the texture map, then it is possible to satisfy two lookup requests with a single read request" (Col. 6, lines 56-61 – emphasis added). Clearly, the **texture mode** indicating a 16-bit per pixel **texture lookup,** as in Rivard, simply fails to support the Examiner's allegation of "sending and receiving instructions to and from DRAM" (emphasis added), as alleged by the Examiner.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 21-22, 25-26 and 29-30, the Examiner has argued the following:

> "Rivard also teaches the arrival of memory data and associated instructions at the cache data store and memory data resolver component of the text cache system is coordinated by pipeline latency elements (col. 7 lines 4-7; memory data has associated instructions; it should be noted that depending on the DRAM design, there could be a constant latency to effect a page hit; however, in the instance when the instructions and

data are combined together, no pipeline latency element will be required; therefore, when the Rivard reference is modified as taught by Wang below, the modified system might have the data and the instructions coordinated when arrived at cache data store and memory data resolver, and will not require the use of pipeline latency element; however, having the pipeline latency element will help to coordinate the instructions and data at their arrival at cache data store and memory data resolver, when there is a delay between the arrival of data and it's associated instructions; it should be noted that Rivard has this pipeline latency element in addition to the claimed limitation, and moreover, the examiner interprets that the modified system with pipeline latency element will still function as effectively and generate the desired results, even when there is no delay between the arrival of the data and it's associated instructions)."

Appellant respectfully disagrees and asserts that Wang merely teaches that "[t]he graphics hardware system 108, over bus 100", supplies data and control signals to a local frame buffer memory 110" (Col. 5, lines 63-67) and that the "texture map data access (TDA) circuit 200… provides an efficient mechanism for processing texture map data requests (in the form of addresses) whereby useful texture map data can be supplied from a cache memory 251 to the filter 260 simultaneously during a fetch interval wherein other texture data is being fetched from main memory 102 or from a local frame buffer 110 (FIG. 1)" (Col. 6, lines 53-61 – emphasis added). Further, Wang teaches that "FIFO memory 240 of the TDA circuit 200" (Col. 8, lines 38-39), "…contains a number of entries, (1)-(m), for storing hit or miss addresses" and that "[i]f the texture map data corresponding to the bottom entry address is not yet available in the cache memory 251, FIFO memory 240 is stalled until the data becomes available" (Col. 8, lines 14-16 and 22-25 – emphasis added). Therefore, Wang teaches that the texture map data access circuit processes texture map data requests and supplies texture map data from a cache memory 251, where a FIFO memory 240 of the texture map data access circuit, which stores hit or miss addresses, stalls if the texture map data corresponding to the bottom entry address is not yet available in the cache memory 251.

Furthermore, Rivard teaches that "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," where "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that "generates memory requests for all misses… and forwards the requests to DRAM 655 for information retrieval" and that the "pipeline latency elements 1025… coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030" (Col. 6, line 40-Col. 7, line 7 – emphasis added). Therefore, Rivard's pipeline latency

elements 1025 coordinate arrival of the requested texel values retrieved from DRAM and the arrival of the associated instructions at cache data store 1030.

However, Wang's texture map data access circuit that processes texture map data requests and supplies texture map data from a cache memory 251, where a FIFO memory 240 of the texture map data access circuit stalls if the **texture map data** corresponding to the bottom entry address is not yet available in the cache memory 251, in addition to Rivard's pipeline latency elements 1025 that coordinate arrival of the requested texel values retrieved from DRAM and the arrival of the associated instructions at cache data store 1030, fails to support the Examiner's allegation that the "when the Rivard reference is modified as taught by Wang below, the modified system might have the data and the instructions coordinated when arrived at cache data store and memory data resolver, and will not require the use of pipeline latency element" (emphasis added), as alleged by the Examiner.

Further, a texture map data access circuit that processes texture map data requests and supplies texture map data from a cache memory 251, as in Wang, in addition a memory request generator 1020 that generates memory requests for all misses and forwards the requests to DRAM 655 for information retrieval, where pipeline latency elements 1025 to coordinating arrival of the requested texel values retrieved from DRAM and the arrival of the associated instructions at cache data store 1030, as in Rivard, simply fails to suggest "sending an instruction request to video memory, where a texture module... sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Clearly, a texture map data access circuit processing texture map data requests, as in Wang, in addition to a memory request generator generating memory requests for texel values for all misses and forwarding the requests to DRAM for texel value retrieval, as in Rivard, simply fails to suggest that "a texture module... sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Additionally, with respect to independent Claims 1, 24, and 27, the Examiner has relied on Figure 6 and Figure 10; Col. 4, lines 46-57; Col. 6, lines 45-67; and Col. 7, lines 1-10 from Rivard (reproduced above) to make a prior art showing of appellant's claimed "receiving instructions from the video memory in response to the instruction request utilizing the texture

module in the graphics pipeline" (see this or similar, but not necessarily identical language in the foregoing independent claims).

Specifically, the Examiner has argued that "DRAM returns memory data to cache data store and memory data resolver component of texel cache system, which further sends this information to the texture mapping stage; [where] the texture mapping stage and the texel cache system together are considered the cache module, so the information/memory data is passed between the components of the texture module."

Appellant respectfully disagrees, and points out that in the second paragraph on Page 7 of the Office Action mailed 03/20/2007, the Examiner has stated that "Rivard does not explicitly teach to combine texture mapping stage and texel cache system to form a texture module," and thus, Rivard simply fails to suggest "receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline" (emphasis added), as claimed by appellant.

In addition, the excerpts from Rivard relied upon by the Examiner merely teach that "DRAM 655 returns the memory data on bus 660 to cache data store and memory data resolver 1030" (emphasis added). However, merely returning memory data to a cache data store, as in Rivard, simply fails to teach "receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline" (emphasis added), as claimed by appellant.

Furthermore, appellant respectfully asserts that Rivard teaches that "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57) Additionally, Rivard discloses that "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015" and "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" which "generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-36, and lines 48-53 – emphasis added). Further, Rivard discloses that

"DRAM 655 returns the memory data on bus 660 to cache data store and memory data resolver 1030, which stores the memory data at the write address" (Col. 6, lines 53-56 – emphasis added).

However, the disclosure of a memory request generator that generates memory requests for all misses and forwards the requests to DRAM for information retrieval, and that the DRAM then returns the memory data which is stored at the write address, as in Rivard, simply fails to even suggest "receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline" (emphasis added), as claimed by appellant. Clearly, returning memory data which is stored at a write address after a cache miss, as in Rivard, fails to meet "receiving instructions from the video memory in response to the instruction request" (emphasis added), as claimed by appellant. Appellant emphasizes that the memory data from the DRAM, as disclosed in Rivard, fails to teach or suggest "instructions," in the manner as claimed by appellant.

Furthermore, in the Office Action mailed 09/20/2007, on Page 7, lines 3-7; Page 5, lines 9-17; Page 6, line 10 to Page 7, line 4; and Page 8, lines 4-8, the Examiner has argued the following:

> "In response to [appellant's] arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986)." (Page 7, lines 3-7)

> "Rivard further teaches receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline (it is interpreted that instructions stored in DRAM that are requested are basically stored as data; and Rivard teaches to send back data in some form in response to the memory request generated as a result of a miss; DRAM returns memory data to cache data store and memory data resolver component of texel cache system, which further sends this information to the texture mapping stage; the texture mapping stage and the texel cache system together are considered as texture module, so the information/memory data is passed between the components of the texture module)." (Page 5, lines 9-17)

> "The examiner further interprets that Rivard also teaches the arrival of memory data and associated instructions at the cache data store and memory data resolver component of the text cache system is coordinated by pipeline latency elements (col. 7 lines 4-7; memory data has associated instructions are returned by DRAM, though the data has it's associated instructions). Therefore, the examiner brings in the Wang reference that suggests a graphics subunit (texture module) in a graphics hardware system that supplies data and control signals to local frame buffer memory for executing a series

of display instructions (Fig. 1, col. 5 lines 38-67; the computer memory includes the local frame buffer memory, which corresponds to the DRAM; based on the control signals send by the graphics hardware system, the frame buffer returns the polygon display instructions that includes the texture data, so that the graphics subunit executes the display instructions using this data). Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device (col. 5 lines 43-45 and lines 63-67)." (Page 6, line 10 to Page 7, line 4)

"However, the examiner interprets that Rivard teaches to send a request to DRAM via a texture module, and based on this request, DRAM sends back data and it's associated information to the texture module (see the arguments above for details). Nonetheless, Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM, though the data has it's associated functions." (Page 8, lines 4-8)

First, appellant notes that the Examiner's arguments on Page 7, lines 5-21; and Page 8, lines 9-20 in the Office Action mailed 09/20/2007 are substantially similar to the arguments made by the Examiner on Page 6, line 10 to Page 7, line 4 excerpted above. Further, on Page 7, line 21 the Examiner has additionally citied "In re Lockhart, 90 USPQ (CCPA 1951)." Second, appellant has clearly responded to such arguments above regarding how the excerpts repeatedly relied upon by the Examiner fail to meet or suggest appellant's claimed "receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline" (emphasis added), as claimed by appellant.

Further, appellant respectfully asserts that the excerpts from Rivard relied upon by the Examiner merely teach that "[m]emory request generator 1020 generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" such that "DRAM 655 returns the memory data on bus 660 to cache data store and memory data resolver 1030" (Col. 6, lines 50-55 – emphasis added). However, Rivard's disclosure that memory requests are forwarded to DRAM for information retrieval, where the DRAM returns the memory data, fails to support the Examiner's allegation that "Rivard further teaches receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline" (emphasis added). Clearly, forwarding memory requests to DRAM for information retrieval, as in Rivard, simply fails to specifically teach "receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline" (emphasis added), as claimed by appellant.

In addition, appellant respectfully asserts that the excerpts from Rivard relied upon by the Examiner teach that "[b]ecause memory request generator 1020 is between cache tag blocks 1010, 1015 and cache data store 1030, generator 1020 can perform DRAM 655 memory requests before the address and instruction information reach cache data store and memory data resolver 1030" (Col. 6, lines 62-66 – emphasis added). Furthermore, Wang teaches "a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" and that "[m]any of the polygon display instructions include texture data to be displayed within the polygon" (Col. 5, lines 40-48 – emphasis added). However, the mere disclosure of a memory request generator 1020 performing DRAM memory requests before the instruction information reaches the cache data store, as in Rivard, in addition to the disclosure of executing display instructions that may include texture data, as in Wang, simply fails to suggest "receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline" (emphasis added), as claimed by appellant. Clearly, performing memory requests before instruction information reaches the cache data store, as in Rivard, in addition to the disclosure of executing display instructions that may include texture data, as in Wang, simply fails to even suggest an "instruction request," much less "receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline" (emphasis added), as claimed by appellant.

Further, appellant respectfully asserts that Rivard's disclosure of a "graphics accelerator system 635 [that] includes pipeline latency elements 1025 to coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030" (Col. 7, lines 4-7) clearly *teaches away* from the Examiner's allegation that it would be obvious to "have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device," as noted by the Examiner, since in Rivard the "pipeline latency elements… coordinate arrival of the memory data and… the associated instructions" (emphasis added). Clearly, as argued above, it would not be obvious from the teachings of Rivard and Wang to "receiv[e] instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline" (emphasis added), as claimed by appellant.

Still yet, appellant agrees with the Examiner's statement that "Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM." For example, appellant asserts that the figures and excerpts from Rivard relied upon by the Examiner teach that texel cache system 650 "includes cache tags 1010, cache tags 1015, a memory request generator 1020, pipeline latency elements 1025 and cache data store and memory data resolver 1030" (see Figure 10; Col. 6, lines 22-26 – emphasis added), where the "[m]emory request generator 1020 generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 50-53 – emphasis added). Clearly, a texel cache system 650 including a memory request generator 1020 that generates memory requests that are forwarded to DRAM 655 for information retrieval, as in Rivard, simply fails to suggest "receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline" (emphasis added), as claimed by appellant.

In addition, the Examiner has admitted that "Rivard does not explicitly teach to combine texture mapping stage and texel cache system to form a texture module," and has argued that "it would have been obvious to one of ordinary skill in the art at the time of present invention to combine texture mapping stage and texel cache system of Rivard to work together as a texture module."

Appellant disagrees and respectfully asserts that it would not have been obvious to combine the texture mapping stage and texel cache system in Rivard, as suggested by the Examiner. Specifically, Rivard teaches the use of "pipeline latency elements 1025" which "coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030" (Col. 7, lines 3-7 - emphasis added). However, Rivard actually *teaches away* from the Examiner's above allegation, in addition to appellant's claim language, by intentionally incorporating the pipeline latency elements to coordinate arrival of the memory data from separate sources. Therefore, for the reasons argued above, it would not be obvious to "combine texture mapping stage and texel cache system," as argued by the Examiner. Thus, it would not have been obvious to "receiv[e] instructions from the video memory in response to the instruction request utilizing the texture module..." (emphasis added), as appellant claims.

Appellant thus formally requests a specific showing of the subject matter in ALL of the claims in any future action. Note excerpt from MPEP below.

"If the appellant traverses such an [Official Notice] assertion the examiner should cite a reference in support of his or her position." See MPEP 2144.03.

Still yet, the Examiner has admitted that "Rivard does not explicitly teach that the memory returns instructions along with data, in response to [the] instruction request from the texture module," but has argued that "Wang teaches a graphics subunit (texture module) in a graphics hardware system that supplies data and control signals to local frame buffer memory for executing a series of display instructions (Fig. 1, col. 5 lines 38-67...)."

Appellant respectfully disagrees. The excerpt from Wang relied on by the Examiner merely discloses a "graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory." However, only generally disclosing that a graphics subunit executes instructions stored in computer memory, as in Wang, fails to specifically meet appellant's claimed "receiving instructions from the video memory in response to the instruction request utilizing the texture module..." (emphasis added), as appellant claims. In fact, appellant notes that Wang only discloses that the "graphics subunit 109 includ[es] a texture engine 10, [and] a polygon engine 12," and that the "texture engine 10 is responsible for retrieving the texture map data," whereas the "polygon engine 12...performs well known polygon rendering functions" (Col. 6, lines 3-49). Thus, Wang clearly does not disclose "receiving instructions...utilizing the texture module..." (emphasis added), as claimed.

Furthermore, in the Office Action mailed 09/20/2007, on Page 5, line 18 to Page 6, line 9, the Examiner has argued the following:

"Although Rivard discloses the limitations as stated above, Rivard does not explicitly teach to combine texture mapping stage and texel cache system to form a texture module. However, it would have been obvious to one of ordinary skill in art at the time of present invention to combine texture mapping stage and texel cache system of Rivard to work together as a texture module. The unity of diversity of parts would depend more upon the choice of the manufacturer, and the convenience and availability of the machines and tools necessary to construct the texture module, than on any inventive concept. One of ordinary skill in art, furthermore, would have expected [appellant's] invention to perform equally well with Rivard's reference that teaches to send and receive information/instruction from the texture mapping stage and texel cache system to the DRAM because using this components together will also result in sending

and receiving instructions to and from DRAM. Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to modify Rivard to obtain the invention as specified in the claim." (Page 5, line 18 to Page 6, line 9)

First, appellant respectfully asserts that the Examiner's arguments on Page 7, lines 8-21 of the Office Action mailed 09/20/2007 are substantially similar to the arguments made by the Examiner on Page 5, line 18 to Page 6, line 9 excerpted above. Further, on Page 7, line 21 the Examiner has additionally citied "*In re Lockhart*, 90 USPQ (CCPA 1951)."

Further, appellant respectfully disagrees with the Examiner's arguments and asserts that it would not be obvious to "combine texture mapping stage and texel cache system of Rivard to work together as a texture module," as alleged by the Examiner. For example, Rivard teaches that a "[g]raphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650" such that "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 49-57 – emphasis added). Clearly, Rivard teaches and suggests the use of graphic pipeline stages, such as a texture mapping block 645 and texel cache system 650, which simply fails to support the Examiner's allegation that it would be obvious to "combine texture mapping stage and texel cache system of Rivard to work together as a texture module." Therefore, Rivard's teachings of a separate texture mapping block 645 and texel cache system 650 simply fails to suggest "receiving instructions...utilizing the texture module..." (emphasis added), as claimed.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 22-23, the Examiner has argued the following:

"In response to [appellant's] arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck* & Co., 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986)."

Appellant respectfully disagrees, and asserts that Wang teaches that "the graphics subunit 109 including a texture engine 10, a polygon engine 12 and a pixel pipeline 16" (Col. 6, lines 3-5 –

emphasis added) and that Fig. 2 illustrates that the texture engine 10 further contains a texture map data access circuit 200, where the "[t]exture map data retrieval (TDA) circuit 200 performs texture map data retrieval processes" (Col. 6, lines 23-25 – emphasis added). Further, Wang teaches that "[t]he host computer system 112 provides data and control signals via bus 100 to a graphics hardware unit or system, e.g., 'graphics card' 108," where "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" (Col. 5, lines 38-43 – emphasis added). In addition, Rivard teaches, as illustrated in Fig. 10, that the texel cache system 650 "includes cache tags 1010, cache tags 1015, a memory request generator 1020, pipeline latency elements 1025 and cache data store and memory data resolver 1030" (Col. 6, lines 22-26), and that "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57 – emphasis added).

However, disclosing that the graphics subunit 109 includes a texture engine 10 that contains a texture map data access circuit 200 that performs a texture map data retrieval processes, in addition to disclosing that the host computer system 112 provides data and control signals via bus 100 to a graphics hardware unit that contains a graphics subunit 109 that executes a series of display instructions found within a display list stored in computer memory, as in Wang, in addition to disclosing that texture mapping block 645 sends information via bus 647 to texel cache system 650 that sends information via bus 649 back to texture mapping block 645, where the texel cache system 650 includes cache, a memory request generator, pipeline latency elements, and cache data store and memory data resolver, as in Rivard, simply fails to suggest "receiving instructions from the video memory in response to the instruction request utilizing the texture module..." (emphasis added), as claimed by appellant. Clearly, Wang's disclosure that the graphics subunit 109 executes a series of display instructions found within a display list stored in computer memory, in addition to Rivard's disclosure that the texture mapping block 645 sends information via bus 647 to texel cache system 650 that further sends information via bus 649 back to texture mapping block 645, simply fails to suggest "receiving instructions... in response to the instruction request" (emphasis added), as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 25 and 29, the Examiner has argued the following:

"Rivard further teaches receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline (the examiner interprets that a DRAM can store instructions as data, and that the memory request retrieves such data from DRAM; Rivard teaches to send back data in some form in response to the memory request generated as a result of a miss; DRAM returns memory data to cache data store and memory data resolver component of texel cache system, which further sends this information to the texture mapping stage; the texture mapping stage and the texel cache system together are considered as texture module, so the information/memory data is passed between the components of the texture module)."

Appellant respectfully disagrees and asserts that Rivard teaches that, as illustrated in Fig. 10, the "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015," where the "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," and that "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that "generates memory requests for all misses..., and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-53 – emphasis added). Additionally, Figs. 6 and 10 clearly illustrate that the texel sample address computation block 1005 of Fig. 10 is a part of the texture mapping stage 645 that sends information via bus 647 to the cache tag blocks 1010 and 1015 of the texel cache system 650.

However, teaching that the texel sample address computation block 1005 of the texture mapping stage 645 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015 via bus 647 (as illustrated in Fig. 10), where if the cache tag blocks of the texel cache system 650 determine that the requested texel values are not stored in cache data store 1030, then the cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020 of the texel cache system 650 that generates memory requests for all misses and forwards the requests to DRAM 655 for information retrieval, as in Rivard, simply fails to suggest "receiving instructions from the video memory in response to the instruction request utilizing the texture module..." (emphasis added), as claimed by appellant. Clearly, the cache tag blocks of the texel cache

system 650 determining that the requested texel values are not stored in cache data store 1030, in addition to the memory request generator 1020 of the texel cache system 650 generating and forwarding memory requests to DRAM 655 for information retrieval, as in Rivard, simply fails to even suggest "receiving instructions... in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 26 and 30-31, the Examiner has argued the following:

> "However, Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM. Therefore, the examiner brings in the Wang reference that suggests a graphics subunit (texture module) in a graphics hardware system that supplies data and control signals to local frame buffer memory for executing a series of display instructions (Fig. 1, col. 5 lines 38-67; the computer memory includes the local frame buffer memory, which corresponds to the DRAM; based on the control signals send by the graphics hardware system, the frame buffer returns the polygon display instructions that includes the texture data, so that the graphics subunit executes the display instructions using this data; it should be noted that Wang teaches to return display instructions from the display list to the graphics hardware system for further processing, where the display instructions include texture data). Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device (col. 5 lines 43-45 and lines 63-67)."

Appellant agrees with the Examiner's statement that "Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM," as stated by the Examiner. Further, appellant respectfully disagrees with the Examiner's assertions made with respect to the Wang reference.

Specifically, appellant asserts that Wang merely teaches that "the graphics subunit 109 includ[es] a texture engine 10, a polygon engine 12 and a pixel pipeline 16" (Col. 6, lines 3-5 – emphasis added) and that Fig. 2 illustrates that the texture engine 10 further contains a texture map data access circuit 200, where the "[t]exture map data retrieval (TDA) circuit 200 performs texture map data retrieval processes" (Col. 6, lines 23-25 – emphasis added). Further, Wang teaches that the "[t]exture data is stored in computer readable (e.g., volatile) memory units of system 112, or local frame buffer 110" (Col. 5, lines 38-51 – emphasis added). Additionally, Wang teaches that

"[t]he host computer system 112 provides data and control signals via bus 100 to a graphics hardware unit or system, e.g., 'graphics card' 108," where "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" (Col. 5, lines 38-43 – emphasis added).

However, the host computer system providing data and control signals via bus 100 to a graphics hardware unit that contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory, in addition to a texture map data retrieval (TDA) circuit of the texture engine performing texture map data retrieval processes, where texture data is stored in memory units of the system 112 or the local frame buffer 110, as in Wang, fails to support the Examiner's assertion that Wang suggests "a graphics subunit (texture module) in a graphics hardware system that supplies data and control signals to local frame buffer memory for executing a series of display instructions," as asserted by the Examiner, and further fails to even suggest "receiving instructions... in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant.

Again, appellant respectfully asserts that at least the first and third elements of the *prima facie* case of obviousness have not been met, since it would be *unobvious* to combine the references, as noted above, and the prior art excerpts, as relied upon by the Examiner, fail to teach or suggest all of the claim limitations, as noted above.

> ### Group #2: Claims 25 and 26

With respect to independent Claims 25 and 26, the Examiner has relied on Figure 6 and Figure 10, in addition to Col. 4, lines 46-57; and Col. 6, line 45-Col. 7, line 10 (excerpted below) from Rivard to make a prior art showing of appellant's claimed "sending an instruction request to video memory, where a texture module sends the instruction request to the video memory" (see this or similar, but not necessarily identical language in the foregoing independent claims).

> "Graphics application program 670 transfers graphical information from data storage 625 into DRAM 655 for local storage. Graphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650. Texture

mapping block 645 sends information via bus 647 to texel cache system
650, and texel cache system 650 sends information via bus 649 back to
texture mapping block 645." (Col. 4, lines 46-57 - emphasis added)

"Cache data store 1030 responsively outputs on lines 649 the texture
values cached in the read location.

If a miss occurs, then cache tag blocks 1010 and 1015 forward a cache
write address to memory request generator 1020. Memory request
generator 1020 generates memory requests for all misses (up to four for
bi-linear sampling and up to eight for tri-linear sampling), and
forwards the requests to DRAM 655 for information retrieval. DRAM 655
returns the memory data on bus 660 to cache data store and memory data
resolver 1030, which stores the memory data at the write address. If
the interface to DRAM 655 is 32-bits wide, the texture mode indicates a
16-bit per pixel texture lookup and the data is conveniently aligned in
the texture map, then it is possible to satisfy two lookup requests
with a single read request. However, if the data is not aligned, then
two read requests are needed.

Because memory request generator 1020 is between cache tag blocks 1010,
1015 and cache data store 1030, generator 1020 can perform DRAM 655
memory requests before the address and instruction information reach
cache data store and memory data resolver 1030. Once memory requests
are generated, there is, depending on the DRAM design, a constant
latency of about five to ten clock cycles (or possibly more) which
includes time for exiting and reentering the graphics pipeline
hardware, to effect a page hit. Therefore, graphics accelerator system
635 includes pipeline latency elements 1025 to coordinate arrival of
the memory data and of the associated instructions at cache data store
and memory data resolver 1030." (Col. 6, line 45-Col. 7, line 10 -
emphasis added)

Appellant respectfully asserts that the figures relied on by the Examiner only generally illustrate a computer system and a block diagram detailing the graphics accelerator system showing the pipeline latency elements 1025. In addition, the excerpts relied on by the Examiner merely teach that the "[m]emory request generator 1020 generates memory requests for all misses (up to four for bi-linear sampling and up to eight for tri-linear sampling), and forwards the requests to DRAM 655 for information retrieval" (emphasis added). However, disclosing that a memory request is generated for all misses, as in Rivard, fails to teach "sending an instruction request to video memory, where a texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Appellant notes that the Examiner has argued that "the texture mapping stage and the texel cache system together are considered the texture module." Appellant respectfully disagrees and asserts that Rivard merely suggests that the "[g]raphics accelerator system 635 includes graphics

pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650" where the "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 49-57 – emphasis added). Clearly, the texture mapping stage is separate from the texture cache system, as clearly disclosed in Rivard, and thus fails to suggest "sending an instruction request to video memory, where a texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

In addition, the Examiner has argued that "memory requests/read requests for all misses that are forwarded to DRAM are instruction requests based on which DRAM sends back information... [where] the DRAM sends memory data based on the memory requests/read requests from the texture module." Further, the Examiner has argued that "[m]emory requests/read requests act as an instruction to DRAM, which performs a particular function based on the request."

Appellant respectfully disagrees, and asserts that Rivard merely discloses that the "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57). Additionally, Rivard discloses that "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015" and "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" which "generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-36, and lines 48-53 – emphasis added).

However, Rivard's disclosure that the memory request generator 1020, which is included in the texel cache system (see Figure 10), generates memory requests for all misses and forwards the requests to DRAM for information retrieval, simply fails to even suggest "sending an instruction request to video memory, where a texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Moreover, the texel sample address computation block (included in texture mapping block 645) which forwards sample points to the

cache tag blocks 1010 and 1015 (included in texel cache system 650), as disclosed in Rivard, fails to meet "sending an instruction request to video memory, where a texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Furthermore, the memory request to the DRAM, as in Rivard, fails to suggest "an instruction request," in the manner as claimed by appellant.

In addition, appellant respectfully asserts that the following excerpts from Rivard further demonstrate that the Rivard fails to disclose appellant's claimed "sending an instruction request to video memory, where a texture module sends the instruction request to the video memory" (see this or similar, but not necessarily identical language in the foregoing independent claims).

> "If the interface to DRAM 655 is 32-bits wide, the texture mode indicates a 16-bit per pixel texture lookup and the data is conveniently aligned in the texture map, then it is possible to satisfy two lookup requests with a single read request. However, if the data is not aligned, then two read requests are needed." (Col. 6, lines 56-61 – emphasis added)

> "Therefore, graphics accelerator system 635 includes pipeline latency elements 1025 to coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030." (Col. 7, lines 3-7 – emphasis added)

> "…the memory request generator is coupled between the cache tag block and the cache data store for performing a memory request before an address and instruction information associated with the needed texels reach the cache data store." (Col. 10, lines 48-52 – emphasis added)

First, appellant respectfully points out that Rivard merely teaches that "the memory request generator is coupled between the cache tag block and the cache data store for performing a memory request before an address and instruction information associated with the needed texels reach the cache data store" (emphasis added). Clearly, coordinating the arrival of "memory data", where "the data is conveniently aligned in the texture map" (emphasis added), as in Rivard (see excerpts above), fails to teach "sending an instruction request" (emphasis added), as claimed by appellant.

Second, appellant respectfully asserts that Rivard simply discloses that a memory request is performed before an address and instruction information associated with the needed texels reach the cache data store (see excerpts above). Thus, the memory data in Rivard simply relates to

texel data, which clearly fails to suggest "sending <u>an instruction request</u> to video memory, where <u>a texture module</u> <u>sends the instruction request</u> to the video memory" (emphasis added), as claimed by appellant.

Furthermore, the Examiner has argued that "the definition of instruction provided by dictionary.com, which defines instructions as a command given to a computer to carry out a particular operation and can contain data to be used in the operation; here DRAM sends memory data based on the memory requests/read requests from the texture module." Further, the Examiner has argued that "[m]emory requests/read requests act as an instruction to DRAM, which performs a particular function based on the request."

Appellant respectfully disagrees with the Examiner's arguments and asserts that the dictionary.com reference provided by the Examiner merely defines an instruction as "a command given to a computer to carry out a particular operation." Appellant respectfully asserts that Rivard merely discloses that the "<u>cache tag blocks 1010 and 1015 forward</u> a cache write address to <u>memory request generator</u> 1020" which "<u>generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval</u>" (Col. 6, lines 33-36, and lines 48-53 – emphasis added). However, the disclosure of the memory request generator generating a memory request and forwarding the request to DRAM <u>for information retrieval</u>, as in Rivard, simply fails to suggest "sending <u>an instruction request</u> to video memory, where <u>a texture module</u> <u>sends the instruction request</u> to the video memory" (emphasis added), as claimed by appellant. Clearly, generating and forwarding <u>a memory request</u> to DRAM, as in Rivard, fails to suggest "sending <u>an instruction request</u>," particularly where "<u>instructions</u> [are received]...in response to the instruction request" (emphasis added), in the context as claimed by appellant.

In the Office Action mailed 09/20/2007, on Page 2, line 19 to Page 3, line 19; and Page 4, lines 3-7, the Examiner has argued the following:

> "However, the examiner interprets that Rivard (Fig. 6, Fig. 10, col. 4 lines 46-57, col. 6 lines 45-67, col. 7 lines 1-10) teaches sending an instruction request to video memory, where a texture module in a graphics pipeline sends the instruction request to the video memory (a texture mapping stage sends information to texel cache system; the texture mapping stage and the texel cache system together are considered as texture module, so the information is passed between the components of the texture module;

texel cache system comprising the memory request generator, pipeline latency elements, and cache data store and memory data resolver forwards the memory requests/read requests to DRAM for information retrieval. [T]he examiner interprets that Rivard discloses sending an instruction request to video memory by generating memory request for all misses, and forwards the requests to DRAM for information retrieval (Fig. 6, Fig. 10, col. 4 lines 46-57, col. 6 lines 45-67, col. 7 lines 1-10). It is interpreted that an instruction request could be considered as merely a request because it is basically a request that requires DRAM to perform necessary operation. The examiner also interprets that instructions stored in DRAM that are requested are basically stored as data. It is further interpreted that memory requests/read requests act as an instruction to DRAM based on which DRAM performs a particular function. Memory requests/read requests for all misses that are forwarded to DRAM are instruction requests based on which DRAM sends back information; the examiner relies on the definition of instruction provided by dictionary.com, which defines instructions as a command given to a computer to carry out a particular operation and can contain data to be used in the operation; here DRAM sends memory based on the memory requests/read requests from the texture module." (Page 2, line 19 to Page 3, line 19)

"In response to [appellant's] arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986)." (Page 4, lines 3-7)

First, appellant notes that the Examiner's arguments on Page 4, line 8 to Page 5, line 8 in the Office Action mailed 09/20/2007 are substantially similar to the arguments made by the Examiner on Page 2, line 19 to Page 3, line 19 excerpted above. Second, appellant has clearly responded to such arguments above regarding how the excerpts repeatedly relied upon by the Examiner fail to meet or suggest appellant's claimed "sending an instruction request to video memory, where a texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Further, appellant respectfully disagrees with the Examiner's arguments and asserts that the figures and excerpts from Rivard relied upon by the Examiner merely disclose that "the texture mode indicates a 16-bit per pixel texture lookup" in addition to "satisfy[ing] two lookup requests with a single read request" (Col. 6, lines 56-61 – emphasis added). Additionally, the excerpts from Rivard teach that "pipeline latency elements 1025... coordinate arrival of the memory data and of the associated instructions" (Col. 7, lines 3-7 – emphasis added) such that "the memory request generator... perform[s] a memory request before an address and instruction information associated with the needed texels reach the cache data store" (Col. 10, lines 48-52 – emphasis

added). Clearly, the texture mode indicating a texture lookup, in addition to the memory request generator performing a memory request, as in Rivard, simply fails to support the Examiner's allegation that "a texture module in a graphics pipeline sends the instruction request to the video memory" (emphasis added), and especially does not teach "sending an instruction request to video memory, where a texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant..

In addition, Rivard's disclosure that the "cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020," which "generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 48-53 – emphasis added). However, the disclosure of the memory request generator generating a memory request and forwarding the request to DRAM for information retrieval, as in Rivard, fails to support the Examiner's allegation "that Rivard discloses sending an instruction request to video memory by generating memory request for all misses, and forwards the requests to DRAM for information retrieval" (emphasis added). Clearly, generating a memory request and forwarding the request to DRAM for information retrieval, as in Rivard, simply fails to even suggest specifically teach "sending an instruction request to video memory, where a texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Still yet, appellant respectfully asserts that the excerpts from Rivard relied upon by the Examiner disclose that the "[g]raphics application program 670 transfers graphical information from data storage 625 into DRAM 655 for local storage" (Col. 4, lines 46-48 – emphasis added). Additionally, the excerpts disclose that "[i]f a hit occurs, then cache tag blocks 1010 and 1015 forward a cache read address through memory request generator 1020 and through pipeline latency elements 1025 to cache data store 1030" such that "[c]ache data store 1030 responsively outputs on lines 649 the texture values cached in the read location" (Col. 6, lines 42-47 – emphasis added). Clearly, Rivard is disclosing the transfer of graphical information into DRAM and outputting cached texture values, which simply fails to support the Examiner's allegation that "instructions stored in DRAM that are requested are basically stored as data" (emphasis added). Therefore, the disclosure of transferring graphical information into DRAM and outputting cached texture values, as in Rivard, simply fails to suggest "sending an instruction

request to video memory, where a texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Furthermore, appellant again asserts that the dictionary.com reference provided by the Examiner merely defines an instruction as "a command given to a computer to carry out a particular operation." In addition, the excerpts from Rivard disclose that the "memory request generator 1020 generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 50-53 – emphasis added). However, generating and forwarding memory requests to DRAM for information retrieval, as in Rivard, fails to support the Examiner's allegation that "[m]emory requests/read requests for all misses that are forwarded to DRAM are instruction requests based on which DRAM sends back information" (emphasis added). Clearly, generating and forwarding memory requests to DRAM for information retrieval, as in Rivard, simply fails to specifically teach "sending an instruction request to video memory, where a texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Appellant emphasizes that the memory request of Rivard fails to meet appellant's claimed "instruction request," as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, on Pages 32-34, the Examiner has stated that the above arguments are similar in scope to the arguments discussed above and to refer to the statements presented above in regards to the reasons provided for the above arguments. Specifically, in the paragraph on Pages 15-16, 16-18, 19-20, 20-21, 23-24 and 27-28, the Examiner has argued the following:

> "However, the examiner interprets that Rivard (Fig. 6, Fig. 10, col. 4 lines 46-57, col. 6 lines 45-67, col. 7 lines 1-10) teaches sending an instruction request to video memory, where a texture module in a graphics pipeline sends the instruction request to the video memory (a texture mapping stage sends information to texel cache system; the texture mapping stage and the texel cache system together are considered as texture module, so the information is passed between the components of the texture module; texel cache system comprising the memory request generator, pipeline latency elements, and cache data store and memory data resolver forwards the memory requests/read requests to DRAM for information retrieval; the examiner interprets that Rivard discloses sending an instruction request to video memory by generating memory request for all misses, and forwards the requests to DRAM for information retrieval). It is interpreted that an instruction request could be considered as merely a request because it is basically a request that requires DRAM to perform necessary operation. The examiner also interprets that a DRAM can store instructions as data, and that the memory request

retrieves such data from DRAM. It is further interpreted that when there is a miss, a memory requests/read requests is send to DRAM, which acts as an instruction to DRAM based on which DRAM performs a particular function of returning data for further processing. Therefore, it is reasonable to interpret that memory requests/read requests for all misses that are forwarded to DRAM are instruction requests based on which DRAM sends back information to the graphics pipeline for further processing (it should be noted that the examiner relies on the definition of instruction provided by dictionary.com, which defines instructions as a command given to a computer to carry out a particular operation and can contain data to be used in the operation; in this case, when the memory/read request is send to DRAM, DRAM sends memory data based on the request to the texture module of the graphics pipeline for further processing)."

Appellant respectfully disagrees and asserts that Rivard teaches that "[g]raphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650," where "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 49-57 – emphasis added).

However, a graphics accelerator system 635 including graphics pipeline stages 640 that includes a texture mapping stage 645, where the texture mapping stage 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645, as in Rivard, simply fails to support the Examiner's assertion that "the texture mapping stage and the texel cache system together are considered as texture module," as alleged by the Examiner. Furthermore, Rivard's teachings that the graphics pipeline stages 640 include a texture mapping stage 645, where the texture mapping stage 645 sends information via bus 647 to texel cache system 650, simply fails to suggest that "a texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Clearly, teaching that the texture mapping block 645, included in the graphics pipeline stages 640, sends information via bus 647 to the separate texel cache system 650, which is **not included** in the graphics pipeline stages 640, as in Rivard, simply fails to even suggest that "a texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Additionally, appellant respectfully asserts that Rivard teaches that, as illustrated in Fig. 10, the "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015," where the "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," and that "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that "generates memory requests for all misses..., and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-53 – emphasis added).

Therefore, Rivard teaches that the texel sample address computation block 1005 of the texture mapping block 645 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015, where if the cache tag blocks determine that the requested texel values are not stored in cache data store 1030, then the cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020 that generates memory requests for all misses and forwards the requests to DRAM 655 for information retrieval. Clearly, generating a memory request when the cache tag blocks determine that the requested texel values are not stored in cache data store, and forwarding the memory request to DRAM for information retrieval, as in Rivard, simply fails to support the Examiner's allegation that "an instruction request could be considered as merely a request because it is basically a request that requires DRAM to perform necessary operation" (emphasis added), as alleged by the Examiner, and further fails to suggest appellant's claimed "instruction request," as claimed by appellant. Clearly, generating a memory request when the requested texel values are not stored in cache data store, as in Rivard, simply fails to even suggest appellant's claimed "instruction request," as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, on Pages 32-34, the Examiner has stated that the above arguments are similar in scope to the arguments discussed above and to refer to the statements presented above in regards to the reasons provided for the above arguments. Specifically, in the paragraph on Pages 18, 24-25 and 28-29, the Examiner has argued the following:

> "The examiner further interprets that Rivard does not explicitly teach texture mapping stage and texel cache system together functions as a texture module. However,

it would have been obvious to one of ordinary skill in art at the time of present invention to combine texture mapping stage and texel cache system of Rivard to work together as a texture module (it should be noted that the examiner interprets texture mapping stage and texel cache system combined together will perform functions similar to the claimed texture module, and therefore it would be obvious to modify the Rivard reference). The unity of diversity of parts would depend more upon the choice of the manufacturer, and the convenience and availability of the machines and tools necessary to construct the texture module, than on any inventive concept. One of ordinary skill in art, furthermore, would have expected [appellant's] invention to perform equally well with Rivard's reference that teaches to send and receive information/instruction from the texture mapping stage and texel cache system to the DRAM because using this components together will also result in sending and receiving instructions to and from DRAM. Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to modify Rivard to obtain the invention as specified in the claim."

First, appellant respectfully agrees with the Examiner's assertion that "Rivard does not explicitly teach texture mapping stage and texel cache system together functions as a texture module," since, as argued hereinabove, Rivard teaches that the texture mapping stage 645, included in the graphics pipeline stages 640, sends information via bus 647 to the separate texel cache system 650, which is **not included** in the graphics pipeline stages 640.

Second, appellant respectfully disagrees with the Examiner's allegation that "it would have been obvious to one of ordinary skill in art at the time of present invention to combine texture mapping stage and texel cache system of Rivard to work together as a texture module," as alleged by the Examiner. Appellant asserts that Rivard teaches that "[a] system for caching texel information in a cache data store, for use in a graphics rendering system," where the "system includes a texel memory storing texel information, ...a first cache data storage..., a second cache data storage..., cache tag blocks..., and a memory request generator..." (Abstract – emphasis added). Further, Rivard teaches, as illustrated in Fig. 10, that the texel cache system 650 "includes cache tags 1010, cache tags 1015, a memory request generator 1020, pipeline latency elements 1025 and cache data store and memory data resolver 1030" (Col. 6, lines 22-26), and that "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57 – emphasis added).

Therefore, Rivard is clearly teaching that the use of bus 647 for sending information from the texture mapping stage 645, included in the graphics pipeline stages 640, to the separate texel

cache system 650, and the use of bus 649 for sending information from the texel cache system 650 back to texture mapping stage 645, which fails to suggest that it would be obvious "to combine texture mapping stage and texel cache system of Rivard to work together as a texture module," as alleged by the Examiner, and further fails to suggest that "a texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Furthermore, appellant thus formally requests a specific showing of the subject matter in ALL of the claims in any future action. Note excerpt from MPEP below.

"If the [appellant] traverses such an [Official Notice] assertion the examiner should cite a reference in support of his or her position." See MPEP 2144.03.

Third, appellant respectfully asserts that Rivard teaches that, as illustrated in Fig. 10, the "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015," where the "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," and that "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that "generates memory requests for all misses..., and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-53 – emphasis added). Additionally, Figs. 6 and 10 clearly illustrate that the texel sample address computation block 1005 of Fig. 10 is a part of the texture mapping stage 645 that sends information via bus 647 to the cache tag blocks 1010 and 1015 of the texel cache system 650.

Therefore, Rivard teaches that the texel sample address computation block 1005 of the texture mapping stage 645 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015 via bus 647 (as illustrated in Fig. 10), where if the cache tag blocks determine that the requested texel values are not stored in cache data store 1030, then the cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020 that generates memory requests for all misses. However, the texture mapping stage 645 forwarding sample points to the cache tag blocks of the texel cache system 650, where the cache tag blocks forward a cache write address to the memory request generator that generates the memory request, as in Rivard, fails to support that

Examiner's allegation that the Rivard "reference that teaches to send and receive information/instruction from the texture mapping stage and texel cache system to the DRAM because using this components together will also result in sending and receiving instructions to and from DRAM" (emphasis added), as alleged by the Examiner, and further fails to suggest that "a texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Furthermore, Rivard teaches that "[i]f the interface to DRAM 655 is 32-bits wide, the texture mode indicates a 16-bit per pixel texture lookup and the data is conveniently aligned in the texture map, then it is possible to satisfy two lookup requests with a single read request" (Col. 6, lines 56-61 – emphasis added). Clearly, the texture mode indicating a 16-bit per pixel texture lookup, as in Rivard, simply fails to support the Examiner's allegation of "sending and receiving instructions to and from DRAM" (emphasis added), as alleged by the Examiner.

In the Examiner's Answer mailed 06/26/2008, on Pages 32-34, the Examiner has stated that the above arguments are similar in scope to the arguments discussed above and to refer to the statements presented above in regards to the reasons provided for the above arguments. Specifically, in the paragraph on Pages 21-22, 25-26 and 29-30, the Examiner has argued the following:

> "Rivard also teaches the arrival of memory data and associated instructions at the cache data store and memory data resolver component of the text cache system is coordinated by pipeline latency elements (col. 7 lines 4-7; memory data has associated instructions; it should be noted that depending on the DRAM design, there could be a constant latency to effect a page hit; however, in the instance when the instructions and data are combined together, no pipeline latency element will be required; therefore, when the Rivard reference is modified as taught by Wang below, the modified system might have the data and the instructions coordinated when arrived at cache data store and memory data resolver, and will not require the use of pipeline latency element; however, having the pipeline latency element will help to coordinate the instructions and data at their arrival at cache data store and memory data resolver, when there is a delay between the arrival of data and it's associated instructions; it should be noted that Rivard has this pipeline latency element in addition to the claimed limitation, and moreover, the examiner interprets that the modified system with pipeline latency element will still function as effectively and generate the desired results, even when there is no delay between the arrival of the data and it's associated instructions)."

Appellant respectfully disagrees and asserts that Wang merely teaches that "[t]he graphics hardware system 108, over bus 100", supplies data and control signals to a local frame buffer memory 110" (Col. 5, lines 63-67) and that the "texture map data access (TDA) circuit 200... provides an efficient mechanism for processing texture map data requests (in the form of addresses) whereby useful texture map data can be supplied from a cache memory 251 to the filter 260 simultaneously during a fetch interval wherein other texture data is being fetched from main memory 102 or from a local frame buffer 110 (FIG. 1)" (Col. 6, lines 53-61 – emphasis added). Further, Wang teaches that "FIFO memory 240 of the TDA circuit 200" (Col. 8, lines 38-39), "...contains a number of entries, (1)-(m), for storing hit or miss addresses" and that "[i]f the texture map data corresponding to the bottom entry address is not yet available in the cache memory 251, FIFO memory 240 is stalled until the data becomes available" (Col. 8, lines 14-16 and 22-25 – emphasis added). Therefore, Wang teaches that the texture map data access circuit processes texture map data requests and supplies texture map data from a cache memory 251, where a FIFO memory 240 of the texture map data access circuit, which stores hit or miss addresses, stalls if the texture map data corresponding to the bottom entry address is not yet available in the cache memory 251.

Furthermore, Rivard teaches that "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," where "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that "generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" and that the "pipeline latency elements 1025... coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030" (Col. 6, line 40-Col. 7, line 7 – emphasis added). Therefore, Rivard's pipeline latency elements 1025 coordinate arrival of the requested texel values retrieved from DRAM and the arrival of the associated instructions at cache data store 1030.

However, Wang's texture map data access circuit that processes texture map data requests and supplies texture map data from a cache memory 251, where a FIFO memory 240 of the texture map data access circuit stalls if the texture map data corresponding to the bottom entry address is not yet available in the cache memory 251, in addition to Rivard's pipeline latency elements 1025 that coordinate arrival of the requested texel values retrieved from DRAM and the arrival

of the associated instructions at cache data store 1030, fails to support the Examiner's allegation that the "when the Rivard reference is modified as taught by Wang below, the modified system might have the data and the instructions coordinated when arrived at cache data store and memory data resolver, and will not require the use of pipeline latency element" (emphasis added), as alleged by the Examiner.

Further, a texture map data access circuit that processes texture map data requests and supplies texture map data from a cache memory 251, as in Wang, in addition a memory request generator 1020 that generates memory requests for all misses and forwards the requests to DRAM 655 for information retrieval, where pipeline latency elements 1025 to coordinating arrival of the requested texel values retrieved from DRAM and the arrival of the associated instructions at cache data store 1030, as in Rivard, simply fails to suggest "sending an instruction request to video memory, where a texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Clearly, a texture map data access circuit processing texture map data requests, as in Wang, in addition to a memory request generator generating memory requests for texel values for all misses and forwarding the requests to DRAM for texel value retrieval, as in Rivard, simply fails to suggest that "a texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Additionally, with respect to independent Claims 25 and 26, the Examiner has relied on Figure 6 and Figure 10; Col. 4, lines 46-57; Col. 6, lines 45-67; and Col. 7, lines 1-10 from Rivard (reproduced above) to make a prior art showing of appellant's claimed "receiving instructions from the video memory in response to the instruction request" (see this or similar, but not necessarily identical language in the foregoing independent claims).

Specifically, the Examiner has argued that "DRAM returns memory data to cache data store and memory data resolver component of texel cache system, which further sends this information to the texture mapping stage; [where] the texture mapping stage and the texel cache system together are considered the cache module, so the information/memory data is passed between the components of the texture module."

Appellant respectfully disagrees, and points out that in the second paragraph on Page 7 of the Office Action mailed 03/20/2007, the Examiner has stated that "Rivard does not explicitly teach to combine texture mapping stage and texel cache system to form a texture module," and thus, Rivard simply fails to suggest "receiving instructions from the video memory in response to the instruction request" (emphasis added), as claimed by appellant.

In addition, the excerpts from Rivard relied upon by the Examiner merely teach that "DRAM 655 returns the memory data on bus 660 to cache data store and memory data resolver 1030" (emphasis added). However, merely returning memory data to a cache data store, as in Rivard, simply fails to teach "receiving instructions from the video memory in response to the instruction request" (emphasis added), as claimed by appellant.

Furthermore, appellant respectfully asserts that Rivard teaches that "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57) Additionally, Rivard discloses that "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015" and "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" which "generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-36, and lines 48-53 – emphasis added). Further, Rivard discloses that "DRAM 655 returns the memory data on bus 660 to cache data store and memory data resolver 1030, which stores the memory data at the write address" (Col. 6, lines 53-56 – emphasis added).

However, the disclosure of a memory request generator that generates memory requests for all misses and forwards the requests to DRAM for information retrieval, and that the DRAM then returns the memory data which is stored at the write address, as in Rivard, simply fails to even suggest "receiving instructions from the video memory in response to the instruction request" (emphasis added), as claimed by appellant. Clearly, returning memory data which is stored at a write address after a cache miss, as in Rivard, fails to meet "receiving instructions from the video memory in response to the instruction request" (emphasis added), as claimed by appellant.

Appellant emphasizes that the <u>memory data</u> from the DRAM, as disclosed in Rivard, fails to teach or suggest "instructions," in the manner as claimed by appellant.

Furthermore, in the Office Action mailed 09/20/2007, on Page 7, lines 3-7; Page 5, lines 9-17; Page 6, line 10 to Page 7, line 4; and Page 8, lines 4-8, the Examiner has argued the following:

"In response to [appellant's] arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986)." (Page 7, lines 3-7)

"Rivard further teaches receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline (it is interpreted that instructions stored in DRAM that are requested are basically stored as data, and Rivard teaches to send back data in some form in response to the memory request generated as a result of a miss; DRAM returns memory data to cache data store and memory data resolver component of texel cache system, which further sends this information to the texture mapping stage; the texture mapping stage and the texel cache system together are considered as texture module, so the information/memory data is passed between the components of the texture module)." (Page 5, lines 9-17)

"The examiner further interprets that Rivard also teaches the arrival of memory data and associated instructions at the cache data store and memory data resolver component of the text cache system is coordinated by pipeline latency elements (col. 7 lines 4-7; memory data has associated instructions are returned by DRAM, though the data has it's associated instructions). Therefore, the examiner brings in the Wang reference that suggests a graphics subunit (texture module) in a graphics hardware system that supplies data and control signals to local frame buffer memory for executing a series of display instructions (Fig. 1, col. 5 lines 38-67; the computer memory includes the local frame buffer memory, which corresponds to the DRAM; based on the control signals send by the graphics hardware system, the frame buffer returns the polygon display instructions that includes the texture data, so that the graphics subunit executes the display instructions using this data). Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device (col. 5 lines 43-45 and lines 63-67)." (Page 6, line 10 to Page 7, line 4)

"However, the examiner interprets that Rivard teaches to send a request to DRAM via a texture module, and based on this request, DRAM sends back data and it's associated information to the texture module (see the arguments above for details). Nonetheless, Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM, though the data has it's associated functions." (Page 8, lines 4-8)

First, appellant notes that the Examiner's arguments on Page 7, lines 5-21; and Page 8, lines 9-20 in the Office Action mailed 09/20/2007 are substantially similar to the arguments made by the Examiner on Page 6, line 10 to Page 7, line 4 excerpted above. Further, on Page 7, line 21 the Examiner has additionally citied "*In re Lockhart*, 90 USPQ (CCPA 1951)." Second, appellant has clearly responded to such arguments above regarding how the excerpts repeatedly relied upon by the Examiner fail to meet or suggest appellant's claimed "receiving instructions from the video memory in response to the instruction request" (emphasis added), as claimed by appellant.

Further, appellant respectfully asserts that the excerpts from Rivard relied upon by the Examiner merely teach that "[m]emory request generator 1020 generates memory requests for all misses… and forwards the requests to DRAM 655 for information retrieval" such that "DRAM 655 returns the memory data on bus 660 to cache data store and memory data resolver 1030" (Col. 6, lines 50-55 – emphasis added). However, Rivard's disclosure that memory requests are forwarded to DRAM for information retrieval, where the DRAM returns the memory data, fails to support the Examiner's allegation that "Rivard further teaches receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline" (emphasis added). Clearly, forwarding memory requests to DRAM for information retrieval, as in Rivard, simply fails to specifically teach "receiving instructions from the video memory in response to the instruction request" (emphasis added), as claimed by appellant.

In addition, appellant respectfully asserts that the excerpts from Rivard relied upon by the Examiner teach that "[b]ecause memory request generator 1020 is between cache tag blocks 1010, 1015 and cache data store 1030, generator 1020 can perform DRAM 655 memory requests before the address and instruction information reach cache data store and memory data resolver 1030" (Col. 6, lines 62-66 – emphasis added). Furthermore, Wang teaches "a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" and that "[m]any of the polygon display instructions include texture data to be displayed within the polygon" (Col. 5, lines 40-48 – emphasis added). However, the mere disclosure of memory request generator 1020 performing DRAM memory requests before the instruction information reaches the cache data store, as in Rivard, in addition to the disclosure of executing display instructions that may include texture data, as in Wang, simply fails to suggest

"receiving instructions from the video memory in response to the instruction request" (emphasis added), as claimed by appellant. Clearly, performing memory requests before instruction information reaches the cache data store, as in Rivard, in addition to the disclosure of executing display instructions that may include texture data, as in Wang, simply fails to even suggest an "instruction request," much less "receiving instructions from the video memory in response to the instruction request" (emphasis added), as claimed by appellant.

Further, appellant respectfully asserts that Rivard's disclosure of a "graphics accelerator system 635 [that] includes pipeline latency elements 1025 to coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030" (Col. 7, lines 4-7) clearly *teaches away* from the Examiner's allegation that it would be obvious to "have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device," as noted by the Examiner, since the "pipeline latency elements... coordinate arrival of the memory data and... the associated instructions" (emphasis added). Clearly, as argued above, it would not be obvious from the teachings of Rivard and Wang to "receiv[e] instructions from the video memory in response to the instruction request" (emphasis added), as claimed by appellant.

Still yet, appellant agrees with the Examiner's statement that "Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM." For example, appellant asserts that the figures and excerpts from Rivard relied upon by the Examiner teach that texel cache system 650 "includes cache tags 1010, cache tags 1015, a memory request generator 1020, pipeline latency elements 1025 and cache data store and memory data resolver 1030" (see Figure 10; Col. 6, lines 22-26 – emphasis added) where the "[m]emory request generator 1020 generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 50-53 – emphasis added). Clearly, a texel cache system 650 including a memory request generator 1020 that generates memory requests that are forwarded to DRAM 655 for information retrieval, as in Rivard, simply fails to suggest "receiving instructions from the video memory in response to the instruction request" (emphasis added), as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, on Pages 32-34, the Examiner has stated that the above arguments are similar in scope to the arguments discussed above and to refer to the statements presented above in regards to the reasons provided for the above arguments. Specifically, in the paragraph on Pages 22-23, the Examiner has argued the following:

> "In response to [appellant's] arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller,* 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck* & Co., 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986)."

Appellant respectfully disagrees, and asserts that Wang teaches that "the graphics subunit 109 including a texture engine 10, a polygon engine 12 and a pixel pipeline 16" (Col. 6, lines 3-5 – emphasis added) and that Fig. 2 illustrates that the texture engine 10 further contains a texture map data access circuit 200, where the "[t]exture map data retrieval (TDA) circuit 200 performs texture map data retrieval processes" (Col. 6, lines 23-25 – emphasis added). Further, Wang teaches that "[t]he host computer system 112 provides data and control signals via bus 100 to a graphics hardware unit or system, e.g., 'graphics card' 108," where "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" (Col. 5, lines 38-43 – emphasis added). In addition, Rivard teaches, as illustrated in Fig. 10, that the texel cache system 650 "includes cache tags 1010, cache tags 1015, a memory request generator 1020, pipeline latency elements 1025 and cache data store and memory data resolver 1030" (Col. 6, lines 22-26), and that "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57 – emphasis added).

However, disclosing that the graphics subunit 109 includes a texture engine 10 that contains a texture map data access circuit 200 that performs a texture map data retrieval processes, in addition to disclosing that the host computer system 112 provides data and control signals via bus 100 to a graphics hardware unit that contains a graphics subunit 109 that executes a series of display instructions found within a display list stored in computer memory, as in Wang, in addition to disclosing that texture mapping block 645 sends information via bus 647 to texel cache system 650 that sends information via bus 649 back to texture mapping block 645, where the texel cache system 650 includes cache, a memory request generator, pipeline latency

elements, and cache data store and memory data resolver, as in Rivard, simply fails to suggest "receiving instructions from the video memory in response to the instruction request" (emphasis added), as claimed by appellant. Clearly, Wang's disclosure that the graphics subunit 109 executes a series of display instructions found within a display list stored in computer memory, in addition to Rivard's disclosure that the texture mapping block 645 sends information via bus 647 to texel cache system 650 that further sends information via bus 649 back to texture mapping block 645, simply fails to suggest "receiving instructions... in response to the instruction request" (emphasis added), as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, on Pages 32-34, the Examiner has stated that the above arguments are similar in scope to the arguments discussed above and to refer to the statements presented above in regards to the reasons provided for the above arguments. Specifically, in the paragraph on Pages 25 and 29, the Examiner has argued the following:

> "Rivard further teaches receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline (the examiner interprets that a DRAM can store instructions as data, and that the memory request retrieves such data from DRAM; Rivard teaches to send back data in some form in response to the memory request generated as a result of a miss; DRAM returns memory data to cache data store and memory data resolver component of texel cache system, which further sends this information to the texture mapping stage; the texture mapping stage and the texel cache system together are considered as texture module, so the information/memory data is passed between the components of the texture module)."

Appellant respectfully disagrees and asserts that Rivard teaches that, as illustrated in Fig. 10, the "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015," where the "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," and that "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that "generates memory requests for all misses..., and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-53 – emphasis added). Additionally, Figs. 6 and 10 clearly illustrate that the texel sample address computation block 1005 of Fig. 10 is a part of the texture mapping stage 645 that sends information via bus 647 to the cache tag blocks 1010 and 1015 of the texel cache system 650.

However, teaching that the <u>texel sample address computation block 1005 of the texture mapping stage 645</u> forwards <u>the higher resolution sample points A-D</u> to cache tag block 1010 and forwards <u>the lower resolution sample points E-H</u> to cache tag block 1015 <u>via bus 647</u> (as illustrated in Fig. 10), where if the cache tag blocks of <u>the texel cache system 650</u> determine that the <u>requested texel values</u> are <u>not</u> stored in cache data store 1030, then the <u>cache tag blocks 1010 and 1015</u> forward a cache write address to <u>memory request generator 1020</u> of <u>the texel cache system 650</u> that generates <u>memory requests</u> for all misses and forwards the requests to DRAM 655 for information retrieval, as in Rivard, simply fails to suggest "receiving <u>instructions</u> from the video memory <u>in response to the instruction request</u>" (emphasis added), as claimed by appellant. Clearly, the cache tag blocks of <u>the texel cache system 650</u> determining that the <u>requested texel values</u> are <u>not</u> stored in cache data store 1030, in addition to the <u>memory request generator 1020</u> of <u>the texel cache system 650</u> generating and forwarding memory requests to DRAM 655 for information retrieval, as in Rivard, simply fails to even suggest "receiving <u>instructions... in response to the instruction request</u>" (emphasis added), as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, on Pages 32-34, the Examiner has stated that the above arguments are similar in scope to the arguments discussed above and to refer to the statements presented above in regards to the reasons provided for the above arguments. Specifically, in the paragraph on Pages 26 and 30-31, the Examiner has argued the following:

"However, Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM. Therefore, the examiner brings in the Wang reference that suggests a graphics subunit (texture module) in a graphics hardware system that supplies data and control signals to local frame buffer memory for executing a series of display instructions (Fig. 1, col. 5 lines 38-67; the computer memory includes the local frame buffer memory, which corresponds to the DRAM; based on the control signals send by the graphics hardware system, the frame buffer returns the polygon display instructions that includes the texture data, so that the graphics subunit executes the display instructions using this data; it should be noted that Wang teaches to return display instructions from the display list to the graphics hardware system for further processing, where the display instructions include texture data). Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device (col. 5 lines 43-45 and lines 63-67)."

Appellant agrees with the Examiner's statement that "Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM," as stated by the Examiner. Further, appellant respectfully disagrees with the Examiner's assertions made with respect to the Wang reference.

Specifically, appellant asserts that Wang merely teaches that "the graphics subunit 109 includ[es] a texture engine 10, a polygon engine 12 and a pixel pipeline 16" (Col. 6, lines 3-5 – emphasis added) and that Fig. 2 illustrates that the texture engine 10 further contains a texture map data access circuit 200, where the "[t]exture map data retrieval (TDA) circuit 200 performs texture map data retrieval processes" (Col. 6, lines 23-25 – emphasis added). Further, Wang teaches that the "[t]exture data is stored in computer readable (e.g., volatile) memory units of system 112, or local frame buffer 110" (Col. 5, lines 38-51 – emphasis added). Additionally, Wang teaches that "[t]he host computer system 112 provides data and control signals via bus 100 to a graphics hardware unit or system, e.g., 'graphics card' 108," where "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" (Col. 5, lines 38-43 – emphasis added).

However, the host computer system providing data and control signals via bus 100 to a graphics hardware unit that contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory, in addition to a texture map data retrieval (TDA) circuit of the texture engine performing texture map data retrieval processes, where texture data is stored in memory units of the system 112 or the local frame buffer 110, as in Wang, fails to support the Examiner's assertion that Wang suggests "a graphics subunit (texture module) in a graphics hardware system that supplies data and control signals to local frame buffer memory for executing a series of display instructions," as asserted by the Examiner, and further fails to even suggest "receiving instructions... in response to the instruction request" (emphasis added), as claimed by appellant.

Again, appellant respectfully asserts that at least the first and third elements of the *prima facie* case of obviousness have not been met, since it would be *unobvious* to combine the references, as noted above, and the prior art excerpts, as relied upon by the Examiner, fail to teach or suggest all of the claim limitations, as noted above.

Group #3: Claim 28

Moreover, with respect to independent Claim 28, the Examiner has relied on Figures 6 and 10; Col. 4, lines 46-57; Col. 6, lines 45-67; and Col. 7, lines 1-10 in Rivard to make a prior art showing of appellant's claimed "sending an instruction request to video memory, where the texture module sends the instruction request to the video memory."

Appellant respectfully asserts that figures from Rivard relied upon by the Examiner only shows a block diagram of a computer system, and that the excerpts relied on by the Examiner simply disclose that "the cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" where "[m]emory request generator 1020 generates memory requests for all misses." Clearly, generating a <u>memory request</u> for all misses, as in Rivard, does not meet appellant's claimed "sending an <u>instruction request</u>" (emphasis added), as claimed.

In the Office Action mailed 03/20/2007, the Examiner has argued that "Rivard teaches this limitation" and to "refer to the rejection of claim 1... regarding sending an instruction request to video memory, where a texture module in a graphics pipeline sends the instruction request to the video memory." Appellant disagrees with the Examiner's arguments and asserts that, for substantially the same reasons as those argued above with respect to at least some of the independent claims, Rivard fails to even suggest "sending an instruction request," as appellant claims. For example, appellant respectfully points out that Rivard merely teaches that "the <u>memory request</u> generator is coupled between the cache tag block and the cache data store for performing a <u>memory request</u> <u>before</u> an address and instruction information associated with the <u>needed texels</u> reach the cache data store" (Col. 10, lines 48-52 - emphasis added). Clearly, performing a <u>memory request before</u> the address and instruction information reach the cache data store, as in Rivard (see excerpts above), fails to teach "sending an <u>instruction request</u>" (emphasis added), as claimed by appellant.

Furthermore, with respect to Claim 28 and the Examiner's reliance upon the rejection of Claim 1, the Examiner has argued that "the definition of instruction provided by dictionary.com, which defines instructions as a command given to a computer to carry out a particular operation and can

contain data to be used in the operation; here DRAM sends memory data based on the memory requests/read requests from the texture module." Further, the Examiner has argued that "[m]emory requests/read requests act as an instruction to DRAM, which performs a particular function based on the request."

Appellant respectfully disagrees with the Examiner's arguments and asserts that the dictionary.com reference provided by the Examiner merely defines an instruction as "a command given to a computer to carry out a particular operation." Appellant respectfully asserts that Rivard merely discloses that the "cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" which "generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-36, and lines 48-53 – emphasis added). However, the disclosure of the memory request generator generating a memory request and forwarding the request to DRAM for information retrieval, as in Rivard, simply fails to suggest "sending an instruction request to video memory, where the texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Clearly, generating and forwarding a memory request to DRAM, as in Rivard, fails to suggest "sending an instruction request" (emphasis added), in the manner as claimed by appellant.

In the Office Action mailed 09/20/2007, on Page 2, line 19 to Page 3, line 19; and Page 4, lines 3-7, the Examiner has argued the following:

> "However, the examiner interprets that Rivard (Fig. 6, Fig. 10, col. 4 lines 46-57, col. 6 lines 45-67, col. 7 lines 1-10) teaches sending an instruction request to video memory, where a texture module in a graphics pipeline sends the instruction request to the video memory (a texture mapping stage sends information to texel cache system; the texture mapping stage and the texel cache system together are considered as texture module, so the information is passed between the components of the texture module; texel cache system comprising the memory request generator, pipeline latency elements, and cache data store and memory data resolver forwards the memory requests/read requests to DRAM for information retrieval. [T]he examiner interprets that Rivard discloses sending an instruction request to video memory by generating memory request for all misses, and forwards the requests to DRAM for information retrieval (Fig. 6, Fig. 10, col. 4 lines 46-57, col. 6 lines 45-67, col. 7 lines 1-10). It is interpreted that an instruction request could be considered as merely a request because it is basically a request that requires DRAM to perform necessary operation. The examiner also interprets that instructions stored in DRAM that are requested are basically stored as data.

It is further interpreted that memory requests/read requests act as an instruction to DRAM based on which DRAM performs a particular function. Memory requests/read requests for all misses that are forwarded to DRAM are instruction requests based on which DRAM sends back information; the examiner relies on the definition of instruction provided by dictionary.com, which defines instructions as a command given to a computer to carry out a particular operation and can contain data to be used in the operation; here DRAM sends memory based on the memory requests/read requests from the texture module." (Page 2, line 19 to Page 3, line 19)

"In response to [appellant's] arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986)." (Page 4, lines 3-7)

First, appellant notes that the Examiner's arguments on Page 4, line 8 to Page 5, line 8 in the Office Action mailed 09/20/2007 are substantially similar to the arguments made by the Examiner on Page 2, line 19 to Page 3, line 19 excerpted above. Second, appellant has clearly responded to such arguments above regarding how the excerpts repeatedly relied upon by the Examiner fail to meet or suggest appellant's claimed "sending an instruction request to video memory, where the texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Further, appellant respectfully disagrees with the Examiner's arguments and asserts that the figures and excerpts from Rivard relied upon by the Examiner merely disclose that "the texture mode indicates a 16-bit per pixel texture lookup" in addition to "satisfy[ing] two lookup requests with a single read request" (Col. 6, lines 56-61 – emphasis added). Additionally, the excerpts from Rivard teach that "pipeline latency elements 1025... coordinate arrival of the memory data and of the associated instructions" (Col. 7, lines 3-7 – emphasis added) such that "the memory request generator... perform[s] a memory request before an address and instruction information associated with the needed texels reach the cache data store" (Col. 10, lines 48-52 – emphasis added). Clearly, the texture mode indicating a texture lookup, in addition to the memory request generator performing a memory request, as in Rivard, simply fails to support the Examiner's allegation that "a texture module in a graphics pipeline sends the instruction request to the video memory" (emphasis added), and especially does not teach "sending an instruction request to video memory, where the texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

In addition, Rivard's disclosure that the "cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020," which "generates memory requests for all misses… and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 48-53 – emphasis added). However, the disclosure of the memory request generator generating a memory request and forwarding the request to DRAM for information retrieval, as in Rivard, fails to support the Examiner's allegation "that Rivard discloses sending an instruction request to video memory by generating memory request for all misses, and forwards the requests to DRAM for information retrieval" (emphasis added). Clearly, generating a memory request and forwarding the request to DRAM for information retrieval, as in Rivard, simply fails to specifically teach "sending an instruction request to video memory, where the texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Still yet, appellant respectfully asserts that the excerpts from Rivard relied upon by the Examiner disclose that the "[g]raphics application program 670 transfers graphical information from data storage 625 into DRAM 655 for local storage" (Col. 4, lines 46-48 – emphasis added). Additionally, the excerpts disclose that "[i]f a hit occurs, then cache tag blocks 1010 and 1015 forward a cache read address through memory request generator 1020 and through pipeline latency elements 1025 to cache data store 1030" such that "[c]ache data store 1030 responsively outputs on lines 649 the texture values cached in the read location" (Col. 6, lines 42-47 – emphasis added). Clearly, Rivard is disclosing the transfer of graphical information into DRAM and outputting cached texture values, which simply fails to support the Examiner's allegation that "instructions stored in DRAM that are requested are basically stored as data" (emphasis added). Therefore, the disclosure of transferring graphical information into DRAM and outputting cached texture values, as in Rivard, simply fails to suggest "sending an instruction request to video memory, where the texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Furthermore, appellant again asserts that the dictionary.com reference provided by the Examiner merely defines an instruction as "a command given to a computer to carry out a particular operation." In addition, the excerpts from Rivard disclose that the "memory request generator 1020 generates memory requests for all misses… and forwards the requests to DRAM 655 for

information retrieval" (Col. 6, lines 50-53 – emphasis added). However, generating and forwarding memory requests to DRAM for information retrieval, as in Rivard, fails to support the Examiner's allegation that "[m]emory requests/read requests for all misses that are forwarded to DRAM are instruction requests based on which DRAM sends back information" (emphasis added). Clearly, generating and forwarding memory requests to DRAM for information retrieval, as in Rivard, simply fails to specifically teach "sending an instruction request to video memory, where the texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Appellant emphasizes that the memory request of Rivard fails to meet appellant's claimed "instruction request," as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, on Page 35, the Examiner has stated that the above arguments are similar in scope to the arguments discussed above and to refer to the statements presented above in regards to the reasons provided for the above arguments. Specifically, in the paragraph on Pages 15-16, 16-18, 19-20, 20-21, 23-24 and 27-28, in addition to Pages 36-37 and 41-42 the Examiner has argued the following:

"However, the examiner interprets that Rivard (Fig. 6, Fig. 10, col. 4 lines 46-57, col. 6 lines 45-67, col. 7 lines 1-10) teaches sending an instruction request to video memory, where a texture module in a graphics pipeline sends the instruction request to the video memory (a texture mapping stage sends information to texel cache system; the texture mapping stage and the texel cache system together are considered as texture module, so the information is passed between the components of the texture module; texel cache system comprising the memory request generator, pipeline latency elements, and cache data store and memory data resolver forwards the memory requests/read requests to DRAM for information retrieval; the examiner interprets that Rivard discloses sending an instruction request to video memory by generating memory request for all misses, and forwards the requests to DRAM for information retrieval). It is interpreted that an instruction request could be considered as merely a request because it is basically a request that requires DRAM to perform necessary operation. The examiner also interprets that a DRAM can store instructions as data, and that the memory request retrieves such data from DRAM. It is further interpreted that when there is a miss, a memory requests/read requests is send to DRAM, which acts as an instruction to DRAM based on which DRAM performs a particular function of returning data for further processing. Therefore, it is reasonable to interpret that memory requests/read requests for all misses that are forwarded to DRAM are instruction requests based on which DRAM sends back information to the graphics pipeline for further processing (it should be noted that the examiner relies on the definition of instruction provided by dictionary.com, which defines instructions as a command given to a computer to carry out a particular operation and can contain data to be used in the operation; in this case, when the

memory/read request is send to DRAM, DRAM sends memory data based on the request to the texture module of the graphics pipeline for further processing)."

Appellant respectfully disagrees and asserts that Rivard teaches that "[g]raphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650," where "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 49-57 – emphasis added).

However, a graphics accelerator system 635 including graphics pipeline stages 640 that includes a texture mapping stage 645, where the texture mapping stage 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645, as in Rivard, simply fails to support the Examiner's assertion that "the texture mapping stage and the texel cache system together are considered as texture module," as alleged by the Examiner. Furthermore, Rivard's teachings that the graphics pipeline stages 640 include a texture mapping stage 645, where the texture mapping stage 645 sends information via bus 647 to texel cache system 650, simply fails to suggest that "the texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Clearly, teaching that the texture mapping block 645, included in the graphics pipeline stages 640, sends information via bus 647 to the separate texel cache system 650, which is not included in the graphics pipeline stages 640, as in Rivard, simply fails to even suggest that "the texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Additionally, appellant respectfully asserts that Rivard teaches that, as illustrated in Fig. 10, the "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015," where the "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," and that "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that

"generates memory requests for all misses..., and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-53 – emphasis added).

Therefore, Rivard teaches that the texel sample address computation block 1005 of the texture mapping block 645 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015, where if the cache tag blocks determine that the requested texel values are not stored in cache data store 1030, then the cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020 that generates memory requests for all misses and forwards the requests to DRAM 655 for information retrieval. Clearly, generating a memory request when the cache tag blocks determine that the requested texel values are not stored in cache data store, and forwarding the memory request to DRAM for information retrieval, as in Rivard, simply fails to support the Examiner's allegation that "an instruction request could be considered as merely a request because it is basically a request that requires DRAM to perform necessary operation" (emphasis added), as alleged by the Examiner, and further fails to suggest appellant's claimed "instruction request," as claimed by appellant. Clearly, generating a memory request when the requested texel values are not stored in cache data store, as in Rivard, simply fails to even suggest appellant's claimed "instruction request," as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, on Page 35, the Examiner has stated that the above arguments are similar in scope to the arguments discussed above and to refer to the statements presented above in regards to the reasons provided for the above arguments. Specifically, in the paragraph on Pages 18, 24-25 and 28-29, in addition to Pages 37-38 and 42, the Examiner has argued the following:

> "The examiner further interprets that Rivard does not explicitly teach texture mapping stage and texel cache system together functions as a texture module. However, it would have been obvious to one of ordinary skill in art at the time of present invention to combine texture mapping stage and texel cache system of Rivard to work together as a texture module (it should be noted that the examiner interprets texture mapping stage and texel cache system combined together will perform functions similar to the claimed texture module, and therefore it would be obvious to modify the Rivard reference). The unity of diversity of parts would depend more upon the choice of the manufacturer, and the convenience and availability of the machines and tools necessary to construct the texture module, than on any inventive concept. One of ordinary skill in art, furthermore, would have expected [appellant's] invention to perform equally well with Rivard's

reference that teaches to send and receive information/instruction from the texture mapping stage and texel cache system to the DRAM because using this components together will also result in sending and receiving instructions to and from DRAM. Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to modify Rivard to obtain the invention as specified in the claim."

First, appellant respectfully agrees with the Examiner's assertion that "Rivard does not explicitly teach texture mapping stage and texel cache system together functions as a texture module," since, as argued hereinabove, Rivard teaches that the texture mapping stage 645, included in the graphics pipeline stages 640, sends information via bus 647 to the separate texel cache system 650, which is **not included** in the graphics pipeline stages 640.

Second, appellant respectfully disagrees with the Examiner's allegation that "it would have been obvious to one of ordinary skill in art at the time of present invention to combine texture mapping stage and texel cache system of Rivard to work together as a texture module," as alleged by the Examiner. Appellant asserts that Rivard teaches that "[a] system for caching texel information in a cache data store, for use in a graphics rendering system," where the "system includes a texel memory storing texel information, ...a first cache data storage..., a second cache data storage..., cache tag blocks..., and a memory request generator..." (Abstract – emphasis added). Further, Rivard teaches, as illustrated in Fig. 10, that the texel cache system 650 "includes cache tags 1010, cache tags 1015, a memory request generator 1020, pipeline latency elements 1025 and cache data store and memory data resolver 1030" (Col. 6, lines 22-26), and that "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57 – emphasis added).

Therefore, Rivard is clearly teaching that the use of bus 647 for sending information from the texture mapping stage 645, included in the graphics pipeline stages 640, to the separate texel cache system 650, and the use of bus 649 for sending information from the texel cache system 650 back to texture mapping stage 645, which fails to suggest that it would be obvious "to combine texture mapping stage and texel cache system of Rivard to work together as a texture module," as alleged by the Examiner, and further fails to suggest that "the texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Furthermore, appellant thus formally requests a specific showing of the subject matter in ALL of the claims in any future action. Note excerpt from MPEP below.

"If the [appellant] traverses such an [Official Notice] assertion the examiner should cite a reference in support of his or her position." See MPEP 2144.03.

Third, appellant respectfully asserts that Rivard teaches that, as illustrated in Fig. 10, the "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015," where the "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," and that "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that "generates memory requests for all misses..., and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-53 – emphasis added). Additionally, Figs. 6 and 10 clearly illustrate that the texel sample address computation block 1005 of Fig. 10 is a part of the texture mapping stage 645 that sends information via bus 647 to the cache tag blocks 1010 and 1015 of the texel cache system 650.

Therefore, Rivard teaches that the texel sample address computation block 1005 of the texture mapping stage 645 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015 via bus 647 (as illustrated in Fig. 10), where if the cache tag blocks determine that the requested texel values are not stored in cache data store 1030, then the cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020 that generates memory requests for all misses. However, the texture mapping stage 645 forwarding sample points to the cache tag blocks of the texel cache system 650, where the cache tag blocks forward a cache write address to the memory request generator that generates the memory request, as in Rivard, fails to support that Examiner's allegation that the Rivard "reference that teaches to send and receive information/instruction from the texture mapping stage and texel cache system to the DRAM because using this components together will also result in sending and receiving instructions to and from DRAM" (emphasis added), as alleged by the Examiner, and further fails to suggest that

"the texture module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Furthermore, Rivard teaches that "[i]f the interface to DRAM 655 is 32-bits wide, the texture mode indicates a 16-bit per pixel texture lookup and the data is conveniently aligned in the texture map, then it is possible to satisfy two lookup requests with a single read request" (Col. 6, lines 56-61 – emphasis added). Clearly, the texture mode indicating a 16-bit per pixel texture lookup, as in Rivard, simply fails to support the Examiner's allegation of "sending and receiving instructions to and from DRAM" (emphasis added), as alleged by the Examiner.

In the Examiner's Answer mailed 06/26/2008, on Page 35, the Examiner has stated that the above arguments are similar in scope to the arguments discussed above and to refer to the statements presented above in regards to the reasons provided for the above arguments. Specifically, in the paragraph on Pages 21-22, 25-26 and 29-30, in addition to Pages 38-39 and 43-44, the Examiner has argued the following:

"Rivard also teaches the arrival of memory data and associated instructions at the cache data store and memory data resolver component of the text cache system is coordinated by pipeline latency elements (col. 7 lines 4-7; memory data has associated instructions; it should be noted that depending on the DRAM design, there could be a constant latency to effect a page hit; however, in the instance when the instructions and data are combined together, no pipeline latency element will be required; therefore, when the Rivard reference is modified as taught by Wang below, the modified system might have the data and the instructions coordinated when arrived at cache data store and memory data resolver, and will not require the use of pipeline latency element; however, having the pipeline latency element will help to coordinate the instructions and data at their arrival at cache data store and memory data resolver, when there is a delay between the arrival of data and it's associated instructions; it should be noted that Rivard has this pipeline latency element in addition to the claimed limitation, and moreover, the examiner interprets that the modified system with pipeline latency element will still function as effectively and generate the desired results, even when there is no delay between the arrival of the data and it's associated instructions)."

Appellant respectfully disagrees and asserts that Wang merely teaches that "[t]he graphics hardware system 108, over bus 100", supplies data and control signals to a local frame buffer memory 110" (Col. 5, lines 63-67) and that the "texture map data access (TDA) circuit 200... provides an efficient mechanism for processing texture map data requests (in the form of addresses) whereby useful texture map data can be supplied from a cache memory 251 to the

filter 260 simultaneously during a fetch interval wherein other texture data is being fetched from main memory 102 or from a local frame buffer 110 (FIG. 1)" (Col. 6, lines 53-61 – emphasis added). Further, Wang teaches that "FIFO memory 240 of the TDA circuit 200" (Col. 8, lines 38-39), "...contains a number of entries, (1)-(m), for storing hit or miss addresses" and that "[i]f the texture map data corresponding to the bottom entry address is not yet available in the cache memory 251, FIFO memory 240 is stalled until the data becomes available" (Col. 8, lines 14-16 and 22-25 – emphasis added). Therefore, Wang teaches that the texture map data access circuit processes texture map data requests and supplies texture map data from a cache memory 251, where a FIFO memory 240 of the texture map data access circuit, which stores hit or miss addresses, stalls if the texture map data corresponding to the bottom entry address is not yet available in the cache memory 251.

Furthermore, Rivard teaches that "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," where "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that "generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" and that the "pipeline latency elements 1025... coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030" (Col. 6, line 40-Col. 7, line 7 – emphasis added). Therefore, Rivard's pipeline latency elements 1025 coordinate arrival of the requested texel values retrieved from DRAM and the arrival of the associated instructions at cache data store 1030.

However, Wang's texture map data access circuit that processes texture map data requests and supplies texture map data from a cache memory 251, where a FIFO memory 240 of the texture map data access circuit stalls if the texture map data corresponding to the bottom entry address is not yet available in the cache memory 251, in addition to Rivard's pipeline latency elements 1025 that coordinate arrival of the requested texel values retrieved from DRAM and the arrival of the associated instructions at cache data store 1030, fails to support the Examiner's allegation that the "when the Rivard reference is modified as taught by Wang below, the modified system might have the data and the instructions coordinated when arrived at cache data store and memory data resolver, and will not require the use of pipeline latency element" (emphasis added), as alleged by the Examiner.

Further, a <u>texture map data access circuit</u> that processes <u>texture map data requests</u> and supplies <u>texture map data</u> <u>from a cache memory 251</u>, as in Wang, in addition <u>a memory request generator</u> <u>1020</u> that generates memory requests for all misses and forwards the requests to DRAM 655 for information retrieval, where pipeline latency elements 1025 to coordinating arrival of <u>the</u> <u>requested texel values</u> <u>retrieved from DRAM</u> and the arrival of the <u>associated instructions</u> at cache data store 1030, as in Rivard, simply fails to suggest "sending <u>an instruction request</u> to video memory, where <u>the texture module</u> <u>sends the instruction request</u> to the video memory" (emphasis added), as claimed by appellant. Clearly, a <u>texture map data access circuit</u> processing <u>texture map data requests</u>, as in Wang, in addition to a <u>memory request generator</u> generating <u>memory requests for texel values</u> for all misses and forwarding the requests to DRAM <u>for texel</u> <u>value retrieval</u>, as in Rivard, simply fails to suggest that "<u>the texture module</u> <u>sends the instruction</u> <u>request</u> to the video memory" (emphasis added), as claimed by appellant.

In addition, with respect to Claim 28, the Examiner has relied on Figures 6 and 10; Col. 4, lines 46-57; Col. 6, lines 45-67; and Col. 7, lines 1-10 from Rivard to make a prior art showing of appellant's claimed "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module."

Specifically, the Examiner has argued that "Rivard... teaches that DRAM returns memory data to cache data store and memory data resolver component of texel cache system, which further sends this information to the texture mapping stage." Further, the Examiner has argued that "the texture mapping stage and the texel cache system together are considered as texture module, so the information is passed between the components of the texture module." In addition, the Examiner has argued that "[t]he texture mapping stage as shown in Fig. 6 also receives data (instructions) from the rasterizer module of the pipeline, and thus the instructions received from the DRAM via the texel cache system are considered to be the additional instructions."

Appellant respectfully disagrees with the Examiner's arguments and asserts that Rivard merely discloses that "DRAM 655 returns the memory data on bus 660 to cache data store and memory data resolver 1030, which <u>stores the memory data at the write address</u>" (Col. 6, lines 53-56 – emphasis added). However, merely <u>storing memory data</u> at the write address, as in Rivard,

simply fails to suggest "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant. Clearly, storing memory data, as in Rivard, fails to suggest "receiving additional instructions," in the manner as claimed by appellant.

Second, in response to the Examiner's argument that "the texture mapping stage and the texel cache system together are considered as texture module," appellant respectfully disagrees and asserts that Rivard merely teaches that "[g]raphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650" where the "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 49-57 – emphasis added). Clearly, the texture mapping stage is separate from the texture cache system, as in Rivard, which fails to suggest "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant.

Third, in response to the Examiner's argument that "Fig. 6 also receives data (instructions) from the rasterizer module of the pipeline, and thus the instructions received from the DRAM via the texel cache system are considered to be the additional instructions," appellant respectfully disagrees and asserts that Rivard merely teaches that "the memory request generator is coupled between the cache tag block and the cache data store for performing a memory request before an address and instruction information associated with the needed texels reach the cache data store" (emphasis added). Clearly, coordinating the arrival of "memory data", where "the data is conveniently aligned in the texture map" (emphasis added), as in Rivard (see excerpts above), fails to teach "receiving additional instructions" (emphasis added), as claimed by appellant. In addition, appellant respectfully asserts that simply nowhere in the description of Figure 6 is there any disclosure that "Fig. 6 also receives data (instructions) from the rasterizer module of the pipeline," as noted by the Examiner. Thus, Rivard simply fails to meet appellant's claimed "receiving additional instructions… utilizing the texture module," as claimed.

Furthermore, in the Office Action mailed 09/20/2007, on Page 9, line 4 to Page 10, line 5, the Examiner has argued the following:

"In response to [appellant's] arguments against references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986).

In this case, the examiner interprets that Rivard teaches a texture mapping stage as shown in Fig. 6 receiving data (instructions) from the rasterizer module of the pipeline and then receiving data (data and it's associated instructions) from the DRAM. The instructions received from the rasterizer module are considered to be the first instructions, and thus the instructions received from the DRAM via the texel cache system are considered to be the additional instructions.

Since Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM, though the data has it's associated instructions, the examiner relies on the Wang reference that suggests a graphics subunit (texture module) in a graphics hardware system that supplies data and control signals to local frame buffer memory for executing a series of display instructions (Fig. 1, col. 5 lines 38-67; the computer memory includes the local frame buffer memory, which corresponds to the DRAM; based on the control signals send by the graphics hardware system, the display list stored in the frame buffer returns the polygon display instructions that includes the texture data, so that the graphics subunit executes the display instructions using this data). Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device (col. 5 lines 43-45 and lines 63-67)." (Page 9, line 4 to Page 10, line 5)

First, appellant has clearly argued above how the excerpts repeatedly relied upon by the Examiner fail to meet or suggest appellant's claimed "receiving <u>additional instructions</u> from the video memory <u>in response to the instruction request</u> utilizing the texture module" (emphasis added), as claimed by appellant.

Further, appellant respectfully asserts that Figure 6 from the Rivard reference merely teaches that "[g]raphics application program 670 transfers graphical information <u>from data storage</u> 625 <u>into DRAM</u> 655 for local storage" and that "[g]raphics accelerator system 635 includes <u>graphics pipeline stages 640</u> including <u>a texture mapping stage 645</u> for <u>mapping texture information</u> to the graphics information received from graphics application program 670 and for <u>maintaining the texel information</u> in a <u>texel cache system 650</u>" (Col. 4, lines 46-54 – emphasis added).

Additionally, Figure 6 teaches that "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57 – emphasis added). However, Rivard's teaching that graphic pipeline stages 640 includes a texture mapping stage 645 for mapping texture information and maintaining the texel information in a texel cache system 650, such that the texel cache system 650 sends information to texture mapping block 645, simply fails to support the Examiner's allegation that "Rivard teaches a texture mapping stage as shown in Fig. 6 receiving data (instructions) from the rasterizer module of the pipeline and then receiving data (data and it's associated instructions) from the DRAM" (emphasis added). Clearly, a texture mapping stage 645 for mapping texture information and maintaining the texel information in a texel cache system 650, such that the texel cache system 650 sends information to texture mapping block 645, as in Rivard, simply fails to even suggest "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant.

Further, appellant respectfully asserts that the excerpts from Rivard relied upon by the Examiner merely teach that "[m]emory request generator 1020 generates memory requests for all misses… and forwards the requests to DRAM 655 for information retrieval" such that "DRAM 655 returns the memory data on bus 660 to cache data store and memory data resolver 1030" (Col. 6, lines 50-55 – emphasis added). Clearly, forwarding memory requests to DRAM for information retrieval, as in Rivard, simply fails to specifically teach "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant.

In addition, appellant respectfully asserts that the excerpts from Rivard relied upon by the Examiner teach that "[b]ecause memory request generator 1020 is between cache tag blocks 1010, 1015 and cache data store 1030, generator 1020 can perform DRAM 655 memory requests before the address and instruction information reach cache data store and memory data resolver 1030" (Col. 6, lines 62-66 – emphasis added). Furthermore, Wang teaches "a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" and that "[m]any of the polygon display instructions include texture data to be displayed within the polygon" (Col. 5, lines 40-48 – emphasis added). However, the mere

disclosure of <u>memory request generator</u> 1020 performing DRAM <u>memory requests</u> <u>before</u> the instruction information reaches the cache data store, as in Rivard, in addition to the disclosure of executing <u>display instructions</u> that may <u>include texture data</u>, as in Wang, simply fails to suggest "receiving <u>additional instructions</u> from the video memory <u>in response to the instruction request</u> utilizing the texture module" (emphasis added), as claimed by appellant. Clearly, performing <u>memory requests</u> <u>before instruction information</u> reaches the cache data store, as in Rivard, in addition to the disclosure of executing <u>display instructions</u> that may <u>include texture data</u>, as in Wang, simply fails to even suggest an "instruction request," much less "receiving <u>additional instructions</u> from the video memory <u>in response to the instruction request</u> utilizing the texture module" (emphasis added), as claimed by appellant.

Further, appellant respectfully asserts that Rivard's disclosure of a "graphics accelerator system 635 [that] includes pipeline latency elements 1025 to coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030" (Col. 7, lines 4-7) clearly *teaches away* from the Examiner's allegation that it would be obvious to "have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device," as noted by the Examiner, since in Rivard the "<u>pipeline latency elements</u>... <u>coordinate arrival</u> of the <u>memory data</u> and... the <u>associated instructions</u>" (emphasis added). Clearly, as argued above, it would not be obvious from the teachings of Rivard and Wang to "receiving <u>additional instructions</u> from the video memory <u>in response to the instruction request</u> utilizing the texture module" (emphasis added), as claimed by appellant.

Still yet, appellant agrees with the Examiner's statement that "Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM." For example, appellant asserts that the figures and excerpts from Rivard relied upon by the Examiner teach that <u>texel cache system 650</u> "includes cache tags 1010, cache tags 1015, a <u>memory request generator 1020</u>, pipeline latency elements 1025 and cache data store and memory data resolver 1030" (see Figure 10; Col. 6, lines 22-26 – emphasis added) where the "<u>[m]emory request generator 1020</u> generates <u>memory requests</u> for all misses... and forwards the requests to DRAM 655 for <u>information retrieval</u>" (Col. 6, lines 50-53 – emphasis added). Clearly, a texel cache system 650 including a memory request generator 1020 that generates <u>memory requests</u> that are forwarded to DRAM

655 for information retrieval, as in Rivard, simply fails to suggest "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 35-36, the Examiner has argued the following:

> "In response to [appellant's] arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See In re Keller, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); In re Merck & Co., 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986)."

Appellant respectfully disagrees, and asserts that Wang teaches that "the graphics subunit 109 including a texture engine 10, a polygon engine 12 and a pixel pipeline 16" (Col. 6, lines 3-5 – emphasis added) and that Fig. 2 illustrates that the texture engine 10 further contains a texture map data access circuit 200, where the "[t]exture map data retrieval (TDA) circuit 200 performs texture map data retrieval processes" (Col. 6, lines 23-25 – emphasis added). Further, Wang teaches that "[t]he host computer system 112 provides data and control signals via bus 100 to a graphics hardware unit or system, e.g., 'graphics card' 108," where "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" (Col. 5, lines 38-43 – emphasis added). In addition, Rivard teaches, as illustrated in Fig. 10, that the texel cache system 650 "includes cache tags 1010, cache tags 1015, a memory request generator 1020, pipeline latency elements 1025 and cache data store and memory data resolver 1030" (Col. 6, lines 22-26), and that "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57 – emphasis added).

However, disclosing that the graphics subunit 109 includes a texture engine 10 that contains a texture map data access circuit 200 that performs a texture map data retrieval processes, in addition to disclosing that the host computer system 112 provides data and control signals via bus 100 to a graphics hardware unit that contains a graphics subunit 109 that executes a series of display instructions found within a display list stored in computer memory, as in Wang, in addition to disclosing that texture mapping block 645 sends information via bus 647 to texel

cache system 650 that sends information via bus 649 back to texture mapping block 645, where the texel cache system 650 includes cache, a memory request generator, pipeline latency elements, and cache data store and memory data resolver, as in Rivard, simply fails to suggest "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant. Clearly, Wang's disclosure that the graphics subunit 109 executes a series of display instructions found within a display list stored in computer memory, in addition to Rivard's disclosure that the texture mapping block 645 sends information via bus 647 to texel cache system 650 that further sends information via bus 649 back to texture mapping block 645, simply fails to suggest "receiving additional instructions... in response to the instruction request" (emphasis added), as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 38 and 43, the Examiner has argued the following:

> "Rivard further teaches receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline (the examiner interprets that a DRAM can store instructions as data, and that the memory request retrieves such data from DRAM; Rivard teaches to send back data in some form in response to the memory request generated as a result of a miss; DRAM returns memory data to cache data store and memory data resolver component of texel cache system, which further sends this information to the texture mapping stage; the texture mapping stage and the texel cache system together are considered as texture module, so the information/memory data is passed between the components of the texture module)."

Appellant respectfully disagrees and asserts that Rivard teaches that, as illustrated in Fig. 10, the "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015," where the "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," and that "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that "generates memory requests for all misses..., and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-53 – emphasis added). Additionally, Figs. 6 and 10 clearly illustrate that the texel sample address computation block 1005 of Fig. 10 is a part of the

texture mapping stage 645 that sends information via bus 647 to the cache tag blocks 1010 and 1015 of the texel cache system 650.

However, teaching that the texel sample address computation block 1005 of the texture mapping stage 645 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015 via bus 647 (as illustrated in Fig. 10), where if the cache tag blocks of the texel cache system 650 determine that the requested texel values are not stored in cache data store 1030, then the cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020 of the texel cache system 650 that generates memory requests for all misses and forwards the requests to DRAM 655 for information retrieval, as in Rivard, simply fails to suggest "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant. Clearly, the cache tag blocks of the texel cache system 650 determining that the requested texel values are not stored in cache data store 1030, in addition to the memory request generator 1020 of the texel cache system 650 generating and forwarding memory requests to DRAM 655 for information retrieval, as in Rivard, simply fails to even suggest "receiving additional instructions... in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 38 and 43, the Examiner has argued the following:

> "Rivard also teaches a texture mapping stage as shown in Fig. 6 receiving data (instructions) from the rasterizer module of the pipeline and then receiving data (instructions) from the DRAM (it should be noted that the data has it's associated instructions; texture mapping stage in the graphics pipeline receives data from the rasterizer module to perform texturing/rendering operation; this data received from the rasterizer module is considered to be primary data). Therefore, the data received from the DRAM via the texel cache system is considered to be the additional data (Wang reference as shown below teaches that the instructions include data)."

Appellant respectfully disagrees and asserts that Rivard teaches that "[g]raphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650" and that

"[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 49-57 – emphasis added). Further, Wang teaches that "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" (Col. 5, lines 40-43 – emphasis added).

However, texture mapping stage 645 mapping texture information to the graphics information received from graphics application program 670 and maintaining the texel information in a texel cache system 650, where texture mapping block 645 sends information to texel cache system 650, and texel cache system 650 sends information back to texture mapping block 645, as in Rivard, in addition to a 3D graphics subunit 109 executing a series of display instructions found within a display list stored in computer memory, as in Wang, simply fails to suggest "receiving additional instructions… in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant. Clearly, a texture mapping stage 645 sending information to texel cache system 650, and the texel cache system 650 sending information back to texture mapping block 645, as in Rivard, simply fails to suggest "receiving additional instructions… in response to the instruction request" (emphasis added), as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 39-40 and 44-45, the Examiner has argued the following:

"However, Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM. Therefore, the examiner brings in the Wang reference that suggests a graphics subunit (texture module) in a graphics hardware system that supplies data and control signals to local frame buffer memory for executing a series of display instructions (Fig. 1, col. 5 lines 38-67; the computer memory includes the local frame buffer memory, which corresponds to the DRAM; based on the control signals send by the graphics hardware system, the frame buffer returns the polygon display instructions that includes the texture data, so that the graphics subunit executes the display instructions using this data; it should be noted that Wang teaches to return display instructions from the display list to the graphics hardware system for further processing, where the display instructions include texture data; it should be noted that the instructions and data returned by the display list to the graphics hardware system is considered additional instructions or data as the Rivard reference already showed that a rasterizer module passes on preliminary instructions or data to the texture module; it should also be noted that the graphics hardware system includes a 3D graphics subunit that includes a

texture engine that is responsible for retrieving the texture map data for the polygon and mapping the texels of the texture data onto the pixels of the polygon; this texture engine of Wang is functional equivalent of the texture module as suggested by Rivard). Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device (col. 5 lines 43-45 and lines 63-67)."

Appellant agrees with the Examiner's statement that "Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM," as stated by the Examiner. Further, appellant respectfully disagrees with the Examiner's assertions made with respect to the Wang reference.

Specifically, appellant asserts that Wang merely teaches that "the graphics subunit 109 includ[es] a texture engine 10, a polygon engine 12 and a pixel pipeline 16" (Col. 6, lines 3-5 – emphasis added) and that Fig. 2 illustrates that the texture engine 10 further contains a texture map data access circuit 200, where the "[t]exture map data retrieval (TDA) circuit 200 performs texture map data retrieval processes" (Col. 6, lines 23-25 – emphasis added). Further, Wang teaches that the "[t]exture data is stored in computer readable (e.g., volatile) memory units of system 112, or local frame buffer 110" (Col. 5, lines 38-51 – emphasis added). Additionally, Wang teaches that "[t]he host computer system 112 provides data and control signals via bus 100 to a graphics hardware unit or system, e.g., 'graphics card' 108," where "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" (Col. 5, lines 38-43 – emphasis added). In addition, Wang teaches that "[t]he [display] instructions define the rendering of several graphic primitives, e.g., individual points, lines, polygons, fills, BIT BLTs (bit block transfers), textures, etc., and graphics commands" (Col. 5, lines 43-46 – emphasis added).

However, the host computer system providing data and control signals via bus 100 to a graphics hardware unit that contains a 3D graphics subunit 109 for executing a series of display instructions, which define the rendering of several graphic primitives, found within a display list stored in computer memory, in addition to a texture map data retrieval (TDA) circuit of the texture engine performing texture map data retrieval processes, where texture data is stored in memory units of the system 112 or the local frame buffer 110, as in Wang, fails to support the Examiner's assertion that Wang suggests "a graphics subunit (texture module) in a graphics

hardware system that <u>supplies data and control signals to local frame buffer memory for executing a series of display instructions,</u>" as asserted by the Examiner, and further fails to even suggest "receiving <u>additional instructions</u>... <u>in response to the instruction request</u> utilizing the texture module" (emphasis added), as claimed by appellant.

Furthermore, a 3D graphics subunit 109 for <u>executing a series of display instructions</u> that <u>define the rendering of several graphic primitives,</u> such as textures, in addition to performing the retrieval process of <u>texture map data</u> <u>stored</u> in <u>the local frame buffer 110,</u> as in Wang, simply fails to suggest "receiving <u>additional instructions</u>... <u>in response to the instruction request</u> utilizing the texture module" (emphasis added), as claimed by appellant. Clearly, executing a series of display instructions that <u>define the rendering of graphic primitives such as textures,</u> in addition to **retrieving texture map data** stored in a local frame buffer, as in Wang, fails to suggest "receiving <u>additional instructions</u>" (emphasis added), as claimed by appellant, since Wang's retrieval process merely retrieves <u>texture map data</u>.

Again, appellant respectfully asserts that at least the first and third elements of the *prima facie* case of obviousness have not been met, since it would be *unobvious* to combine the references, as noted above, and the prior art excerpts, as relied upon by the Examiner, fail to teach or suggest <u>all</u> of the claim limitations, as noted above.

Group #4: Claim 30

With respect to independent Claim 30, the Examiner has relied on Figure 6 and Figure 10, in addition to Col. 4, lines 46-57; and Col. 6, line 45-Col. 7, line 10 (excerpted below) from Rivard to make a prior art showing of appellant's claimed "sending an instruction request to video memory in a graphics pipeline, where a cache in the graphics pipeline sends the instruction request to the video memory."

> "Graphics application program 670 transfers graphical information from data storage 625 into DRAM 655 for local storage. Graphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650. <u>Texture mapping block 645 sends information via bus 647 to texel cache system</u>

650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645." (Col. 4, lines 46-57 - emphasis added)

"Cache data store 1030 responsively outputs on lines 649 the texture values cached in the read location.

If a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020. Memory request generator 1020 generates memory requests for all misses (up to four for bi-linear sampling and up to eight for tri-linear sampling), and forwards the requests to DRAM 655 for information retrieval. DRAM 655 returns the memory data on bus 660 to cache data store and memory data resolver 1030, which stores the memory data at the write address. If the interface to DRAM 655 is 32-bits wide, the texture mode indicates a 16-bit per pixel texture lookup and the data is conveniently aligned in the texture map, then it is possible to satisfy two lookup requests with a single read request. However, if the data is not aligned, then two read requests are needed.

Because memory request generator 1020 is between cache tag blocks 1010, 1015 and cache data store 1030, generator 1020 can perform DRAM 655 memory requests before the address and instruction information reach cache data store and memory data resolver 1030. Once memory requests are generated, there is, depending on the DRAM design, a constant latency of about five to ten clock cycles (or possibly more) which includes time for exiting and reentering the graphics pipeline hardware, to effect a page hit. Therefore, graphics accelerator system 635 includes pipeline latency elements 1025 to coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030." (Col. 6, line 45-Col. 7, line 10 - emphasis added)

Appellant respectfully asserts that the figures relied on by the Examiner only generally illustrate a computer system and a block diagram detailing the graphics accelerator system showing the pipeline latency elements 1025. In addition, the excerpts relied on by the Examiner merely teach that the "[m]emory request generator 1020 generates memory requests for all misses (up to four for bi-linear sampling and up to eight for tri-linear sampling), and forwards the requests to DRAM 655 for information retrieval" (emphasis added). However, disclosing that a memory request is generated for all misses, as in Rivard, fails to teach "sending an instruction request to video memory in a graphics pipeline, where a cache in the graphics pipeline sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Appellant notes that the Examiner has argued that "the texture mapping stage and the texel cache system together are considered the texture module." Appellant respectfully disagrees and asserts that Rivard merely suggests that the "[g]raphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the

graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650" where the "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 49-57 – emphasis added). Clearly, the texture mapping stage is separate from the texture cache system, as clearly disclosed in Rivard, and thus fails to suggest "sending an instruction request to video memory in a graphics pipeline, where a cache in the graphics pipeline sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

In addition, the Examiner has argued that "memory requests/read requests for all misses that are forwarded to DRAM are instruction requests based on which DRAM sends back information… [where] the DRAM sends memory data based on the memory requests/read requests from the texture module." Further, the Examiner has argued that "[m]emory requests/read requests act as an instruction to DRAM, which performs a particular function based on the request."

Appellant respectfully disagrees, and asserts that Rivard merely discloses that the "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57). Additionally, Rivard discloses that "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015" and "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" which "generates memory requests for all misses… and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-36, and lines 48-53 – emphasis added).

However, Rivard's disclosure that the memory request generator 1020, which is included in the texel cache system (see Figure 10), generates memory requests for all misses and forwards the requests to DRAM for information retrieval, simply fails to even suggest "sending an instruction request to video memory in a graphics pipeline, where a cache in the graphics pipeline sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Moreover, the texel sample address computation block (included in texture mapping block 645) which forwards sample points to the cache tag blocks 1010 and 1015 (included in texel cache system

650), as disclosed in Rivard, fails to meet "sending <u>an instruction request</u> to video memory in a graphics pipeline, where <u>a cache</u> in the graphics pipeline <u>sends the instruction request</u> to the video memory" (emphasis added), as claimed by appellant. Furthermore, the <u>memory request</u> to the DRAM, as in Rivard, fails to suggest "an instruction request," in the manner as claimed by appellant.

In addition, appellant respectfully asserts that the following excerpts from Rivard further demonstrate that the Rivard fails to disclose appellant's claimed "sending <u>an instruction request</u> to video memory in a graphics pipeline, where <u>a cache</u> in the graphics pipeline <u>sends the instruction request</u> to the video memory."

> "If the interface to DRAM 655 is 32-bits wide, the texture mode indicates a <u>16-bit per pixel texture lookup</u> and the data is conveniently aligned in the <u>texture map</u>, then it is possible to satisfy two lookup requests with a single read request. However, if the data is not aligned, then two read requests are needed." (Col. 6, lines 56-61 – emphasis added)

> "Therefore, graphics accelerator system 635 includes <u>pipeline latency</u> elements 1025 to coordinate arrival of the <u>memory data</u> and of the <u>associated instructions</u> at cache data store and memory data resolver 1030." (Col. 7, lines 3-7 – emphasis added).

> "…the memory request generator is coupled between the cache tag block and the cache data store for performing a <u>memory request before an address and instruction information associated with the needed texels reach the cache data store.</u>" (Col. 10, lines 48-52 – emphasis added)

First, appellant respectfully points out that Rivard merely teaches that "the memory request generator is coupled between the cache tag block and the cache data store for performing a memory request <u>before</u> an address and instruction information associated with the needed texels reach the cache data store" (emphasis added). Clearly, coordinating the arrival of "<u>memory data</u>", where "the data is conveniently aligned in the <u>texture map</u>" (emphasis added), as in Rivard (see excerpts above), fails to teach "sending an <u>instruction request</u>" (emphasis added), as claimed by appellant.

Second, appellant respectfully asserts that Rivard simply discloses that a <u>memory request</u> is performed <u>before</u> an address and instruction information associated with the needed <u>texels</u> reach the cache data store (see excerpts above). Thus, the memory data in Rivard simply relates to

texel data, which clearly fails to suggest "sending an instruction request to video memory in a graphics pipeline, where a cache in the graphics pipeline sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Furthermore, the Examiner has argued that "the definition of instruction provided by dictionary.com, which defines instructions as a command given to a computer to carry out a particular operation and can contain data to be used in the operation; here DRAM sends memory data based on the memory requests/read requests from the texture module." Further, the Examiner has argued that "[m]emory requests/read requests act as an instruction to DRAM, which performs a particular function based on the request."

Appellant respectfully disagrees with the Examiner's arguments and asserts that the dictionary.com reference provided by the Examiner merely defines an instruction as "a command given to a computer to carry out a particular operation." Appellant respectfully asserts that Rivard merely discloses that the "cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" which "generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-36, and lines 48-53 – emphasis added). However, the disclosure of the memory request generator generating a memory request and forwarding the request to DRAM for information retrieval, as in Rivard, simply fails to suggest "sending an instruction request to video memory in a graphics pipeline, where a cache in the graphics pipeline sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Clearly, generating and forwarding a memory request to DRAM, as in Rivard, fails to suggest "sending an instruction request," particularly where "instructions [are received]...in response to the instruction request" (emphasis added), in the context as claimed by appellant.

In the Office Action mailed 09/20/2007, on Page 2, line 19 to Page 3, line 19; and Page 4, lines 3-7, the Examiner has argued the following:

> "However, the examiner interprets that Rivard (Fig. 6, Fig. 10, col. 4 lines 46-57, col. 6 lines 45-67, col. 7 lines 1-10) teaches sending an instruction request to video memory, where a texture module in a graphics pipeline sends the instruction request to the video memory (a texture mapping stage sends information to texel cache system; the texture mapping stage and the texel cache system together are considered as texture

module, so the information is passed between the components of the texture module; texel cache system comprising the memory request generator, pipeline latency elements, and cache data store and memory data resolver forwards the memory requests/read requests to DRAM for information retrieval. [T]he examiner interprets that Rivard discloses sending an instruction request to video memory by generating memory request for all misses, and forwards the requests to DRAM for information retrieval (Fig. 6, Fig. 10, col. 4 lines 46-57, col. 6 lines 45-67, col. 7 lines 1-10). It is interpreted that an instruction request could be considered as merely a request because it is basically a request that requires DRAM to perform necessary operation. The examiner also interprets that instructions stored in DRAM that are requested are basically stored as data. It is further interpreted that memory requests/read requests act as an instruction to DRAM based on which DRAM performs a particular function. Memory requests/read requests for all misses that are forwarded to DRAM are instruction requests based on which DRAM sends back information; the examiner relies on the definition of instruction provided by dictionary.com, which defines instructions as a command given to a computer to carry out a particular operation and can contain data to be used in the operation; here DRAM sends memory based on the memory requests/read requests from the texture module." (Page 2, line 19 to Page 3, line 19)

"In response to [appellant's] arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986)." (Page 4, lines 3-7)

First, appellant notes that the Examiner's arguments on Page 4, line 8 to Page 5, line 8 in the Office Action mailed 09/20/2007 are substantially similar to the arguments made by the Examiner on Page 2, line 19 to Page 3, line 19 excerpted above. Second, appellant has clearly responded to such arguments above regarding how the excerpts repeatedly relied upon by the Examiner fail to meet or suggest appellant's claimed "sending an instruction request to video memory in a graphics pipeline, where a cache in the graphics pipeline sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Further, appellant respectfully disagrees with the Examiner's arguments and asserts that the figures and excerpts from Rivard relied upon by the Examiner merely disclose that "the texture mode indicates a 16-bit per pixel texture lookup" in addition to "satisfy[ing] two lookup requests with a single read request" (Col. 6, lines 56-61 – emphasis added). Additionally, the excerpts from Rivard teach that "pipeline latency elements 1025... coordinate arrival of the memory data and of the associated instructions" (Col. 7, lines 3-7 – emphasis added) such that "the memory request generator... perform[s] a memory request before an address and instruction information

associated with the needed texels reach the cache data store" (Col. 10, lines 48-52 – emphasis added). Clearly, the texture mode indicating a texture lookup, in addition to the memory request generator performing a memory request, as in Rivard, simply fails to support the Examiner's allegation that "a texture module in a graphics pipeline sends the instruction request to the video memory" (emphasis added), and especially does not teach "sending an instruction request to video memory in a graphics pipeline, where a cache in the graphics pipeline sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

In addition, Rivard's disclosure that the "cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020," which "generates memory requests for all misses… and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 48-53 – emphasis added). However, the disclosure of the memory request generator generating a memory request and forwarding the request to DRAM for information retrieval, as in Rivard, fails to support the Examiner's allegation "that Rivard discloses sending an instruction request to video memory by generating memory request for all misses, and forwards the requests to DRAM for information retrieval" (emphasis added). Clearly, generating a memory request and forwarding the request to DRAM for information retrieval, as in Rivard, simply fails to specifically teach "sending an instruction request to video memory in a graphics pipeline, where a cache in the graphics pipeline sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Still yet, appellant respectfully asserts that the excerpts from Rivard relied upon by the Examiner disclose that the "[g]raphics application program 670 transfers graphical information from data storage 625 into DRAM 655 for local storage" (Col. 4, lines 46-48 – emphasis added). Additionally, the excerpts disclose that "[i]f a hit occurs, then cache tag blocks 1010 and 1015 forward a cache read address through memory request generator 1020 and through pipeline latency elements 1025 to cache data store 1030" such that "[c]ache data store 1030 responsively outputs on lines 649 the texture values cached in the read location" (Col. 6, lines 42-47 – emphasis added). Clearly, Rivard is disclosing the transfer of graphical information into DRAM and outputting cached texture values, which simply fails to support the Examiner's allegation that "instructions stored in DRAM that are requested are basically stored as data" (emphasis added). Therefore, the disclosure of transferring graphical information into DRAM and

outputting cached <u>texture values,</u> as in Rivard, simply fails to suggest "sending <u>an instruction</u> <u>request</u> to video memory in a graphics pipeline, where <u>a cache</u> in the graphics pipeline <u>sends the</u> <u>instruction request</u> to the video memory" (emphasis added), as claimed by appellant.

Furthermore, appellant again asserts that the dictionary.com reference provided by the Examiner merely defines an instruction as "a command given to a computer to carry out a particular operation." In addition, the excerpts from Rivard disclose that the "<u>memory request generator</u> 1020 <u>generates memory requests</u> for all misses... <u>and forwards the requests to DRAM</u> 655 for <u>information retrieval</u>" (Col. 6, lines 50-53 – emphasis added). However, generating and forwarding <u>memory requests</u> to DRAM for <u>information retrieval,</u> as in Rivard, fails to support the Examiner's allegation that "[m]emory requests/read requests for all misses that are forwarded to DRAM are <u>instruction requests based on which DRAM sends back information</u>" (emphasis added). Clearly, generating and forwarding <u>memory requests</u> to DRAM for <u>information</u> <u>retrieval,</u> as in Rivard, simply fails to specifically teach "sending <u>an instruction request</u> to video memory in a graphics pipeline, where <u>a cache</u> in the graphics pipeline <u>sends the instruction</u> <u>request</u> to the video memory" (emphasis added), as claimed by appellant. Appellant emphasizes that the <u>memory request</u> of Rivard fails to meet appellant's claimed "instruction request," as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 46-47, 47-48, 50-51, 53-54 and 57-58, the Examiner has argued the following:

> "However, the examiner interprets that Rivard (Fig. 6, Fig. 10, col. 4 lines 46-57, col. 6 lines 45-67, col. 7 lines 1-10) teaches sending an instruction request to video memory, where a cache (texel cache system of the graphics pipeline) in a graphics pipeline sends the instruction request to the video memory (texel cache system comprising the memory request generator, pipeline latency elements, and cache data store and memory data resolver forwards the memory requests/read requests to DRAM for information retrieval; the examiner interprets that Rivard discloses sending an instruction request to video memory by generating memory request for all misses, and forwards the requests to DRAM for information retrieval). It is interpreted that an instruction request could be considered as merely a request because it is basically a request that requires DRAM to perform necessary operation. The examiner also interprets that a DRAM can store instructions as data, and that the memory request retrieves such data from DRAM. It is further interpreted that when there is a miss, a memory requests/read requests is send to DRAM, which acts as an instruction to DRAM based on which DRAM performs a particular function of returning data for further processing. Therefore, it is reasonable to

interpret that memory requests/read requests for all misses that are forwarded to DRAM are instruction requests based on which DRAM sends back information to the texel cache system of the graphics pipeline for further processing (it should be noted that the examiner relies on the definition of instruction provided by dictionary.com, which defines instructions as a command given to a computer to carry out a particular operation and can contain data to be used in the operation; in this case, when the memory/read request is send to DRAM, DRAM sends memory data based on the request to the texel cache system of the graphics pipeline for further processing)."

Appellant respectfully disagrees and asserts that Rivard teaches that "[g]raphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650," where "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 49-57 – emphasis added).

However, a graphics accelerator system 635 including graphics pipeline stages 640 that includes a texture mapping stage 645, where the texture mapping stage 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645, as in Rivard, simply fails to support the Examiner's assertion that "the texture mapping stage and the texel cache system together are considered as texture module," as alleged by the Examiner. Furthermore, Rivard's teachings that the graphics pipeline stages 640 include a texture mapping stage 645, where the texture mapping stage 645 sends information via bus 647 to texel cache system 650, simply fails to suggest that "a cache in the graphics pipeline sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Clearly, teaching that the texture mapping block 645, included in the graphics pipeline stages 640, sends information via bus 647 to the separate texel cache system 650, which is not included in the graphics pipeline stages 640, as in Rivard, simply fails to even suggest that "a cache in the graphics pipeline sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Additionally, appellant respectfully asserts that Rivard teaches that, as illustrated in Fig. 10, the "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag

block 1015," where the "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," and that "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that "generates memory requests for all misses..., and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-53 – emphasis added).

Therefore, Rivard teaches that the texel sample address computation block 1005 of the texture mapping block 645 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015, where if the cache tag blocks determine that the requested texel values are not stored in cache data store 1030, then the cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020 that generates memory requests for all misses and forwards the requests to DRAM 655 for information retrieval. Clearly, generating a memory request when the cache tag blocks determine that the requested texel values are not stored in cache data store, and forwarding the memory request to DRAM for information retrieval, as in Rivard, simply fails to support the Examiner's allegation that "an instruction request could be considered as merely a request because it is basically a request that requires DRAM to perform necessary operation" (emphasis added), as alleged by the Examiner, and further fails to suggest appellant's claimed "instruction request," as claimed by appellant. Clearly, generating a memory request when the requested texel values are not stored in cache data store, as in Rivard, simply fails to even suggest appellant's claimed "instruction request," as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, on Page 47, the Examiner has stated that the above arguments are similar in scope to the arguments discussed above and to refer to the statements presented above in regards to the reasons provided for the above arguments. Specifically, in the paragraph on Pages 18, 24-25 and 28-29, the Examiner has argued the following:

> "The examiner further interprets that Rivard does not explicitly teach texture mapping stage and texel cache system together functions as a texture module. However, it would have been obvious to one of ordinary skill in art at the time of present invention to combine texture mapping stage and texel cache system of Rivard to work together as a texture module (it should be noted that the examiner interprets texture mapping stage and texel cache system combined together will perform functions similar to the claimed

texture module, and therefore it would be obvious to modify the Rivard reference). The unity of diversity of parts would depend more upon the choice of the manufacturer, and the convenience and availability of the machines and tools necessary to construct the texture module, than on any inventive concept. One of ordinary skill in art, furthermore, would have expected [appellant's] invention to perform equally well with Rivard's reference that teaches to send and receive information/instruction from the texture mapping stage and texel cache system to the DRAM because using this components together will also result in sending and receiving instructions to and from DRAM. Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to modify Rivard to obtain the invention as specified in the claim."

First, appellant respectfully agrees with the Examiner's assertion that "Rivard does not explicitly teach texture mapping stage and texel cache system together functions as a texture module," since, as argued hereinabove, Rivard teaches that the texture mapping stage 645, included in the graphics pipeline stages 640, sends information via bus 647 to the separate texel cache system 650, which is **not included** in the graphics pipeline stages 640.

Second, appellant respectfully disagrees with the Examiner's allegation that "it would have been obvious to one of ordinary skill in art at the time of present invention to combine texture mapping stage and texel cache system of Rivard to work together as a texture module," as alleged by the Examiner. Appellant asserts that Rivard teaches that "[a] system for caching texel information in a cache data store, for use in a graphics rendering system," where the "system includes a texel memory storing texel information, ...a first cache data storage..., a second cache data storage..., cache tag blocks..., and a memory request generator..." (Abstract – emphasis added). Further, Rivard teaches, as illustrated in Fig. 10, that the texel cache system 650 "includes cache tags 1010, cache tags 1015, a memory request generator 1020, pipeline latency elements 1025 and cache data store and memory data resolver 1030" (Col. 6, lines 22-26), and that "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57 – emphasis added).

Therefore, Rivard is clearly teaching that the use of bus 647 for sending information from the texture mapping stage 645, included in the graphics pipeline stages 640, to the separate texel cache system 650, and the use of bus 649 for sending information from the texel cache system 650 back to texture mapping stage 645, which fails to suggest that it would be obvious "to combine texture mapping stage and texel cache system of Rivard to work together as a texture

module," as alleged by the Examiner, and further fails to suggest that "a cache in the graphics pipeline sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Furthermore, appellant thus formally requests a specific showing of the subject matter in ALL of the claims in any future action. Note excerpt from MPEP below.

"If the [appellant] traverses such an [Official Notice] assertion the examiner should cite a reference in support of his or her position." See MPEP 2144.03.

Third, appellant respectfully asserts that Rivard teaches that, as illustrated in Fig. 10, the "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015," where the "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," and that "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that "generates memory requests for all misses..., and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-53 – emphasis added). Additionally, Figs. 6 and 10 clearly illustrate that the texel sample address computation block 1005 of Fig. 10 is a part of the texture mapping stage 645 that sends information via bus 647 to the cache tag blocks 1010 and 1015 of the texel cache system 650.

Therefore, Rivard teaches that the texel sample address computation block 1005 of the texture mapping stage 645 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015 via bus 647 (as illustrated in Fig. 10), where if the cache tag blocks determine that the requested texel values are not stored in cache data store 1030, then the cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020 that generates memory requests for all misses. However, the texture mapping stage 645 forwarding sample points to the cache tag blocks of the texel cache system 650, where the cache tag blocks forward a cache write address to the memory request generator that generates the memory request, as in Rivard, fails to support that Examiner's allegation that the Rivard "reference that teaches to send and receive information/instruction from the texture mapping stage and texel cache system to the DRAM because using this components together will also result in sending and receiving instructions to

and from DRAM" (emphasis added), as alleged by the Examiner, and further fails to suggest that "a cache in the graphics pipeline sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Furthermore, Rivard teaches that "[i]f the interface to DRAM 655 is 32-bits wide, the texture mode indicates a 16-bit per pixel texture lookup and the data is conveniently aligned in the texture map, then it is possible to satisfy two lookup requests with a single read request" (Col. 6, lines 56-61 – emphasis added). Clearly, the texture mode indicating a 16-bit per pixel texture lookup, as in Rivard, simply fails to support the Examiner's allegation of "sending and receiving instructions to and from DRAM" (emphasis added), as alleged by the Examiner.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 49, 51-52 and 55, the Examiner has argued the following:

> "Rivard also teaches the arrival of memory data and associated instructions at the cache data store and memory data resolver component of the text cache system is coordinated by pipeline latency elements (col. 7 lines 4-7; memory data has associated instructions; it should be noted that depending on the DRAM design, there could be a constant latency to effect a page hit; however, in the instance when the instructions and data are combined together, no pipeline latency element will be required; therefore, when the Rivard reference is modified as taught by Wang below, the modified system might have the data and the instructions coordinated when arrived at cache data store and memory data resolver, and will not require the use of pipeline latency element; however, having the pipeline latency element will help to coordinate the instructions and data at their arrival at cache data store and memory data resolver, when there is a delay between the arrival of data and it's associated instructions; it should be noted that Rivard has this pipeline latency element in addition to the claimed limitation, and moreover, the examiner interprets that the modified system with pipeline latency element will still function as effectively and generate the desired results, even when there is no delay between the arrival of the data and it's associated instructions)."

Appellant respectfully disagrees and asserts that Wang merely teaches that "[t]he graphics hardware system 108, over bus 100", supplies data and control signals to a local frame buffer memory 110" (Col. 5, lines 63-67) and that the "texture map data access (TDA) circuit 200... provides an efficient mechanism for processing texture map data requests (in the form of addresses) whereby useful texture map data can be supplied from a cache memory 251 to the filter 260 simultaneously during a fetch interval wherein other texture data is being fetched from main memory 102 or from a local frame buffer 110 (FIG. 1)" (Col. 6, lines 53-61 – emphasis

added). Further, Wang teaches that "FIFO memory 240 of the TDA circuit 200" (Col. 8, lines 38-39), "...contains a number of entries, (1)-(m), for storing hit or miss addresses" and that "[i]f the texture map data corresponding to the bottom entry address is not yet available in the cache memory 251, FIFO memory 240 is stalled until the data becomes available" (Col. 8, lines 14-16 and 22-25 – emphasis added). Therefore, Wang teaches that the texture map data access circuit processes texture map data requests and supplies texture map data from a cache memory 251, where a FIFO memory 240 of the texture map data access circuit, which stores hit or miss addresses, stalls if the texture map data corresponding to the bottom entry address is not yet available in the cache memory 251.

Furthermore, Rivard teaches that "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," where "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that "generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" and that the "pipeline latency elements 1025... coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030" (Col. 6, line 40-Col. 7, line 7 – emphasis added). Therefore, Rivard's pipeline latency elements 1025 coordinate arrival of the requested texel values retrieved from DRAM and the arrival of the associated instructions at cache data store 1030.

However, Wang's texture map data access circuit that processes texture map data requests and supplies texture map data from a cache memory 251, where a FIFO memory 240 of the texture map data access circuit stalls if the texture map data corresponding to the bottom entry address is not yet available in the cache memory 251, in addition to Rivard's pipeline latency elements 1025 that coordinate arrival of the requested texel values retrieved from DRAM and the arrival of the associated instructions at cache data store 1030, fails to support the Examiner's allegation that the "when the Rivard reference is modified as taught by Wang below, the modified system might have the data and the instructions coordinated when arrived at cache data store and memory data resolver, and will not require the use of pipeline latency element" (emphasis added), as alleged by the Examiner.

Further, a texture map data access circuit that processes texture map data requests and supplies texture map data from a cache memory 251, as in Wang, in addition a memory request generator 1020 that generates memory requests for all misses and forwards the requests to DRAM 655 for information retrieval, where pipeline latency elements 1025 to coordinating arrival of the requested texel values retrieved from DRAM and the arrival of the associated instructions at cache data store 1030, as in Rivard, simply fails to suggest "sending an instruction request to video memory in a graphics pipeline, where a cache in the graphics pipeline sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Clearly, a texture map data access circuit processing texture map data requests, as in Wang, in addition to a memory request generator generating memory requests for texel values for all misses and forwarding the requests to DRAM for texel value retrieval, as in Rivard, simply fails to suggest that "a cache in the graphics pipeline sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Additionally, with respect to independent Claim 30, the Examiner has relied on Figure 6 and Figure 10; Col. 4, lines 46-57; Col. 6, lines 45-67; and Col. 7, lines 1-10 from Rivard (reproduced above) to make a prior art showing of appellant's claimed "receiving instructions from the video memory in response to the instruction request for storage in the cache in the graphics pipeline."

The excerpts from Rivard relied upon by the Examiner merely teach that "DRAM 655 returns the memory data on bus 660 to cache data store and memory data resolver 1030" (emphasis added). However, merely returning memory data to a cache data store, as in Rivard, simply fails to teach "receiving instructions from the video memory in response to the instruction request for storage in the cache in the graphics pipeline" (emphasis added), as claimed by appellant.

Furthermore, appellant respectfully asserts that Rivard teaches that "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57) Additionally, Rivard discloses that "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015" and "[i]f a miss occurs, then cache tag blocks 1010 and

1015 forward a cache write address to memory request generator 1020" which "generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-36, and lines 48-53 – emphasis added). Further, Rivard discloses that "DRAM 655 returns the memory data on bus 660 to cache data store and memory data resolver 1030, which stores the memory data at the write address" (Col. 6, lines 53-56 – emphasis added).

However, the disclosure of a memory request generator that generates memory requests for all misses and forwards the requests to DRAM for information retrieval, and that the DRAM then returns the memory data which is stored at the write address, as in Rivard, simply fails to even suggest "receiving instructions from the video memory in response to the instruction request for storage in the cache in the graphics pipeline" (emphasis added), as claimed by appellant. Clearly, returning memory data which is stored at a write address after a cache miss, as in Rivard, fails to meet "receiving instructions from the video memory in response to the instruction request for storage in the cache" (emphasis added), as claimed by appellant. Appellant emphasizes that the memory data from the DRAM, as disclosed in Rivard, fails to teach or suggest "instructions," in the manner as claimed by appellant.

Furthermore, in the Office Action mailed 09/20/2007, on Page 7, lines 3-7; Page 5, lines 9-17; Page 6, line 10 to Page 7, line 4; and Page 8, lines 4-8; the Examiner has argued the following:

> "In response to [appellant's] arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986)." (Page 7, lines 3-7)

> "Rivard further teaches receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline (it is interpreted that instructions stored in DRAM that are requested are basically stored as data, and Rivard teaches to send back data in some form in response to the memory request generated as a result of a miss; DRAM returns memory data to cache data store and memory data resolver component of texel cache system, which further sends this information to the texture mapping stage; the texture mapping stage and the texel cache system together are considered as texture module, so the information/memory data is passed between the components of the texture module)." (Page 5, lines 9-17)

> "The examiner further interprets that Rivard also teaches the arrival of memory data and associated instructions at the cache data store and memory data resolver

component of the text cache system is coordinated by pipeline latency elements (col. 7 lines 4-7; memory data has associated instructions are returned by DRAM, though the data has it's associated instructions). Therefore, the examiner brings in the Wang reference that suggests a graphics subunit (texture module) in a graphics hardware system that supplies data and control signals to local frame buffer memory for executing a series of display instructions (Fig. 1, col. 5 lines 38-67; the computer memory includes the local frame buffer memory, which corresponds to the DRAM; based on the control signals send by the graphics hardware system, the frame buffer returns the polygon display instructions that includes the texture data, so that the graphics subunit executes the display instructions using this data). Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device (col. 5 lines 43-45 and lines 63-67)." (Page 6, line 10 to Page 7, line 4)

"However, the examiner interprets that Rivard teaches to send a request to DRAM via a texture module, and based on this request, DRAM sends back data and it's associated information to the texture module (see the arguments above for details). Nonetheless, Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM, though the data has it's associated functions." (Page 8, lines 4-8)

First, appellant notes that the Examiner's arguments on Page 7, lines 5-21; and Page 8, lines 9-20 in the Office Action mailed 09/20/2007 are substantially similar to the arguments made by the Examiner on Page 6, line 10 to Page 7, line 4 excerpted above. Further, on Page 7, line 21 the Examiner has additionally cited "*In re Lockhart*, 90 USPQ (CCPA 1951)." Second, appellant has clearly responded to such arguments above regarding how the excerpts repeatedly relied upon by the Examiner fail to meet or suggest appellant's claimed "receiving instructions from the video memory in response to the instruction request for storage in the cache in the graphics pipeline" (emphasis added), as claimed by appellant.

Further, appellant respectfully asserts that the excerpts from Rivard relied upon by the Examiner merely teach that "[m]emory request generator 1020 generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" such that "DRAM 655 returns the memory data on bus 660 to cache data store and memory data resolver 1030" (Col. 6, lines 50-55 – emphasis added). However, Rivard's disclosure that memory requests are forwarded to DRAM for information retrieval, where the DRAM returns the memory data, fails to support the Examiner's allegation that "Rivard further teaches receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline" (emphasis added). Clearly, forwarding memory requests to DRAM for information retrieval, as

in Rivard, simply fails to specifically teach "receiving instructions from the video memory in response to the instruction request for storage in the cache in the graphics pipeline" (emphasis added), as claimed by appellant.

In addition, appellant respectfully asserts that the excerpts from Rivard relied upon by the Examiner teach that "[b]ecause memory request generator 1020 is between cache tag blocks 1010, 1015 and cache data store 1030, generator 1020 can perform DRAM 655 memory requests before the address and instruction information reach cache data store and memory data resolver 1030" (Col. 6, lines 62-66 – emphasis added). Furthermore, Wang teaches "a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" and that "[m]any of the polygon display instructions include texture data to be displayed within the polygon" (Col. 5, lines 40-48 – emphasis added). However, the mere disclosure of memory request generator 1020 performing DRAM memory requests before the instruction information reaches the cache data store, as in Rivard, in addition to the disclosure of executing display instructions that may include texture data, as in Wang, simply fails to suggest "receiving instructions from the video memory in response to the instruction request for storage in the cache in the graphics pipeline" (emphasis added), as claimed by appellant. Clearly, performing memory requests before instruction information reaches the cache data store, as in Rivard, in addition to the disclosure of executing display instructions that may include texture data, as in Wang, simply fails to even suggest an "instruction request," much less "receiving instructions from the video memory in response to the instruction request for storage in the cache in the graphics pipeline" (emphasis added), as claimed by appellant.

Further, appellant respectfully asserts that Rivard's disclosure of a "graphics accelerator system 635 [that] includes pipeline latency elements 1025 to coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030" (Col. 7, lines 4-7) clearly *teaches away* from the Examiner's allegation that it would be obvious to "have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device," as noted by the Examiner, since the "pipeline latency elements... coordinate arrival of the memory data and... the associated instructions" (emphasis added). Clearly, as argued above, it would not be obvious from the teachings of Rivard and Wang to "receiv[e] instructions from the

video memory in response to the instruction request for storage in the cache in the graphics pipeline" (emphasis added), as claimed by appellant.

Still yet, appellant agrees with the Examiner's statement that "Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM." For example, appellant asserts that the figures and excerpts from Rivard relied upon by the Examiner teach that texel cache system 650 "includes cache tags 1010, cache tags 1015, a memory request generator 1020, pipeline latency elements 1025 and cache data store and memory data resolver 1030" (see Figure 10; Col. 6, lines 22-26 – emphasis added) where the "[m]emory request generator 1020 generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 50-53 – emphasis added). Clearly, a texel cache system 650 including a memory request generator 1020 that generates memory requests that are forwarded to DRAM 655 for information retrieval, as in Rivard, simply fails to suggest "receiving instructions from the video memory in response to the instruction request for storage in the cache in the graphics pipeline" (emphasis added), as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Page 50, the Examiner has argued the following:

> "In response to [appellant's] arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986)."

Appellant respectfully disagrees, and asserts that Wang teaches that "the graphics subunit 109 including a texture engine 10, a polygon engine 12 and a pixel pipeline 16" (Col. 6, lines 3-5 – emphasis added) and that Fig. 2 illustrates that the texture engine 10 further contains a texture map data access circuit 200, where the "[t]exture map data retrieval (TDA) circuit 200 performs texture map data retrieval processes" (Col. 6, lines 23-25 – emphasis added). Further, Wang teaches that "[t]he host computer system 112 provides data and control signals via bus 100 to a graphics hardware unit or system, e.g., 'graphics card' 108," where "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" (Col. 5, lines 38-43 – emphasis added). In addition, Rivard teaches, as illustrated in Fig. 10, that the texel cache system 650 "includes

cache tags 1010, cache tags 1015, a memory request generator 1020, pipeline latency elements 1025 and cache data store and memory data resolver 1030" (Col. 6, lines 22-26), and that "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57 – emphasis added).

However, disclosing that the graphics subunit 109 includes a texture engine 10 that contains a texture map data access circuit 200 that performs a texture map data retrieval processes, in addition to disclosing that the host computer system 112 provides data and control signals via bus 100 to a graphics hardware unit that contains a graphics subunit 109 that executes a series of display instructions found within a display list stored in computer memory, as in Wang, in addition to disclosing that texture mapping block 645 sends information via bus 647 to texel cache system 650 that sends information via bus 649 back to texture mapping block 645, where the texel cache system 650 includes cache, a memory request generator, pipeline latency elements, and cache data store and memory data resolver, as in Rivard, simply fails to suggest "receiving instructions from the video memory in response to the instruction request for storage in the cache in the graphics pipeline" (emphasis added), as claimed by appellant. Clearly, Wang's disclosure that the graphics subunit 109 executes a series of display instructions found within a display list stored in computer memory, in addition to Rivard's disclosure that the texture mapping block 645 sends information via bus 647 to texel cache system 650 that further sends information via bus 649 back to texture mapping block 645, simply fails to suggest "receiving instructions... in response to the instruction request for storage in the cache in the graphics pipeline" (emphasis added), as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 51 and 54-55, the Examiner has argued the following:

> "Rivard further teaches receiving instructions from the video memory in response to the instruction request for storage in the cache in the graphics pipeline (the examiner interprets that a DRAM can store instructions as data, and that the memory request retrieves such data from DRAM; Rivard teaches to send back data to the texel cache system in some form in response to the memory request generated as a result of a miss; DRAM returns memory data to cache data store and memory data resolver component of texel cache system, which stores this information and further sends it to the texture mapping stage)."

Appellant respectfully disagrees and asserts that Rivard teaches that, as illustrated in Fig. 10, the "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015," where the "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," and that "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that "generates memory requests for all misses..., and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-53 – emphasis added). Additionally, Figs. 6 and 10 clearly illustrate that the texel sample address computation block 1005 of Fig. 10 is a part of the texture mapping stage 645 that sends information via bus 647 to the cache tag blocks 1010 and 1015 of the texel cache system 650.

However, teaching that the texel sample address computation block 1005 of the texture mapping stage 645 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015 via bus 647 (as illustrated in Fig. 10), where if the cache tag blocks of the texel cache system 650 determine that the requested texel values are not stored in cache data store 1030, then the cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020 of the texel cache system 650 that generates memory requests for all misses and forwards the requests to DRAM 655 for information retrieval, as in Rivard, simply fails to suggest "receiving instructions from the video memory in response to the instruction request for storage in the cache in the graphics pipeline" (emphasis added), as claimed by appellant. Clearly, the cache tag blocks of the texel cache system 650 determining that the requested texel values are not stored in cache data store 1030, in addition to the memory request generator 1020 of the texel cache system 650 generating and forwarding memory requests to DRAM 655 for information retrieval, as in Rivard, simply fails to even suggest "receiving instructions... in response to the instruction request for storage in the cache" (emphasis added), as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 52-53 and 55-56, the Examiner has argued the following:

"However, Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM. Therefore, the examiner brings in the Wang reference that suggests a graphics subunit including a texture map data retrieval circuit that includes a cache control and cache memory unit (cache) in a graphics hardware system that supplies data and control signals to local frame buffer memory for executing a series of display instructions (Fig. 1, col. 5 lines 38-67, col. 7 lines 43-47; the computer memory includes the local frame buffer memory, which corresponds to the DRAM; based on the control signals send by the graphics hardware system, the frame buffer returns the polygon display instructions that includes the texture data, so that the graphics subunit executes the display instructions using this data; it should be noted that Wang teaches to return display instructions from the display list to the graphics hardware system for further processing, where the display instructions include texture data; it should be noted that cache controller circuit 250 of the texture map retrieval circuit fetches the required texture map data from the local frame buffer for storage in cache memory 251). Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device (col. 5 lines 43-45 and lines 63-67)."

Appellant agrees with the Examiner's statement that "Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM," as stated by the Examiner. Further, appellant respectfully disagrees with the Examiner's assertions made with respect to the Wang reference.

Specifically, appellant asserts that Wang merely teaches that "the graphics subunit 109 includ[es] a texture engine 10, a polygon engine 12 and a pixel pipeline 16" (Col. 6, lines 3-5 – emphasis added) and that Fig. 2 illustrates that the texture engine 10 further contains a texture map data access circuit 200, where the "[t]exture map data retrieval (TDA) circuit 200 performs texture map data retrieval processes" (Col. 6, lines 23-25 – emphasis added). Further, Wang teaches that the "[t]exture data is stored in computer readable (e.g., volatile) memory units of system 112, or local frame buffer 110" (Col. 5, lines 38-51 – emphasis added). Additionally, Wang teaches that "[t]he host computer system 112 provides data and control signals via bus 100 to a graphics hardware unit or system, e.g., 'graphics card' 108," where "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" (Col. 5, lines 38-43 – emphasis added). In addition, Wang teaches that "[t]he [display] instructions define the rendering of several graphic primitives, e.g., individual points, lines, polygons, fills, BIT BLTs (bit block transfers), textures, etc., and graphics commands" (Col. 5, lines 43-46 – emphasis added).

However, the host computer system providing data and control signals via bus 100 to a graphics hardware unit that contains a 3D graphics subunit 109 for executing a series of display instructions, which define the rendering of several graphic primitives, found within a display list stored in computer memory, in addition to a texture map data retrieval (TDA) circuit of the texture engine performing texture map data retrieval processes, where texture data is stored in memory units of the system 112 or the local frame buffer 110, as in Wang, fails to support the Examiner's assertion that Wang suggests "a graphics subunit (texture module) in a graphics hardware system that supplies data and control signals to local frame buffer memory for executing a series of display instructions," as asserted by the Examiner, and further fails to even suggest "receiving instructions... in response to the instruction request for storage in the cache" (emphasis added), as claimed by appellant.

Furthermore, a 3D graphics subunit 109 for executing a series of display instructions that define the rendering of several graphic primitives, such as textures, in addition to performing the retrieval process of texture map data stored in the local frame buffer 110, as in Wang, simply fails to suggest "receiving instructions... in response to the instruction request for storage in the cache" (emphasis added), as claimed by appellant. Clearly, executing a series of display instructions that define the rendering of graphic primitives such as textures, in addition to retrieving texture map data stored in a local frame buffer, as in Wang, fails to suggest "receiving instructions" (emphasis added), as claimed by appellant, since Wang's retrieval process merely retrieves texture map data.

Again, appellant respectfully asserts that at least the first and third elements of the *prima facie* case of obviousness have not been met, since it would be *unobvious* to combine the references, as noted above, and the prior art excerpts, as relied upon by the Examiner, fail to teach or suggest all of the claim limitations, as noted above.

Group #5: Claim 18

With respect to Claim 18, the Examiner has relied on Col. 5, lines 43-67; and Col. 6, lines 1-47 from Wang to make a prior art showing of appellant's claimed technique "wherein a complete

instruction set is received in response to the instruction request." Specifically, the Examiner has argued that "Wang... teaches a graphics subunit (texture module) of a graphics hardware system executing a series of display instructions (set of instructions) stored in computer memory" and that "[t]he graphics subunit receives display instructions including texture data based on the supplied data and control signals."

Appellant respectfully disagrees and asserts that Wang merely discloses that "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory," where "[t]he display list generally contains instructions regarding the rendering of several graphic primitives, e.g., individual points, lines, polygons, fills, BIT BLTs (bit block transfers), textures, etc." (Col. 5, lines 40-46 – emphasis added). However, only generally disclosing the graphics subunit executing a series of display instructions found within a display list stored in computer memory, as in Wang, does not specifically teach that "a complete instruction set is received in response to the instruction request," where the "instructions [are received] from the video memory" (emphasis added), in the context as claimed by appellant (see Claim 1 for context).

In fact, appellant notes that the Examiner has relied on the same disclosure in Wang of a "series of display instructions" to meet both appellant's claimed "complete instruction set [that] is received in response to the instruction request" (see Claim 18) and "partial instruction set [that] is received in response to the instruction request" (see Claim 19), as claimed. Clearly, a generally disclosure of a series of instructions, as in Wang, simply cannot meet appellant's claimed "complete instruction set" (Claim 18) and "partial instruction set" (Claim 19), as claimed.

Furthermore, in the Office Action mailed 09/20/2007, on Page 11, line 10 to Page 12, line 8, the Examiner has argued the following:

> "Regarding claim 18, Rivard does not explicitly teach a complete instruction set is received in response to the instruction request. However, Wang (col. 5 lines 43-67, col. 6 lines 1-47) teaches a graphics subunit (texture module) of a graphics hardware system executing a series of display instructions (set of instructions) stored in computer memory (Fig. 1, col. 5 lines 38-67; the computer memory includes the local frame buffer memory, which corresponds to the DRAM; based on the control signals send by the graphics

hardware system, the display list stored in the frame buffer returns the polygon display instructions that include the texture data, so that the graphics subunit executes the display instructions using this data). The graphics subunit receives display instructions including texture data based on the supplied data and control signals (the examiner interprets that the claims do not specify how many instructions are needed to make a set complete; the examiner interprets that if the required operation can be performed from just the one received instruction, then it is considered as a complete set of instruction; the single display instruction received by the graphics from the sub-routine process in response to the control signals correspond to a complete set of instruction; the instruction in the sub-routine is executed by the graphics subunit for rendering the concerned graphics primitive, and therefore it is considered as a complete set of instruction). Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to have the memory return a complete set of instructions as taught by Wang to the texture module of Rivard because this instruction set is needed to render the concerned graphics primitive (col. 5 lines 43-45 and lines 63-67)." (Page 11, line 10 to Page 12, line 8)

Appellant agrees with the Examiner's argument that "Rivard does not explicitly teach a complete instruction set is received in response to the instruction request." Further, appellant respectfully disagrees with the Examiner's arguments with respect to Wang and asserts that the excerpts from Wang relied upon by the Examiner merely disclose that "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" where "[t]he display list generally contains instructions regarding the rendering of several graphic primitives, e.g., individual points, lines, polygons, fills, BIT BLTs (bit block transfers), textures, etc." (Col. 5, lines 40-46 – emphasis added). Further, the excerpts disclose that "[m]any of the polygon display instructions include texture data to be displayed within the polygon" (Col. 5, lines 46-48).

However, appellant asserts that Wang's disclosure of "a 3D graphics subunit 109 for executing a series of display instructions" (emphasis added), clearly fails to support the Examiner's allegation that "if the required operation can be performed from just the one received instruction, then it is considered as a complete set of instruction" (emphasis added). Clearly, a series of display instructions fails to support "just the one received instruction" (emphasis added), as alleged by the Examiner. Furthermore, Wang's disclosure of a series of display instructions simply fails to meet appellant's claimed "complete instruction set," much less disclose that "a complete instruction set is received in response to the instruction request," where the "instructions [are received] from the video memory" (emphasis added), in the context as claimed by appellant (see Claim 1 for context). Therefore, appellant asserts that it would not have been

obvious to "have the memory return a complete set of instructions as taught by Wang to the texture module of Rivard because this instruction set is needed to render the concerned graphics primitive," as alleged by the Examiner.

In the Examiner's Answer mailed 06/26/2008, on Pages 58-59, the Examiner has argued the following:

> "However, the examiner interprets that Rivard does not explicitly teach a complete instruction set is received in response to the instruction request. Therefore, the examiner brings in the Wang reference (col. 5 lines 43-67, col. 6 lines 1-47) that teaches a graphics subunit (texture module) of a graphics hardware system executing a series of display instructions (set of instructions) stored in computer memory (Fig. 1, col. 5 lines 38-67; the computer memory includes the local frame buffer memory, which corresponds to the DRAM; based on the control signals send by the graphics hardware system, the display list stored in the frame buffer returns the polygon display instructions that includes the texture data, so that the graphics subunit executes the display instructions using this data). The graphics subunit receives display instructions including texture data based on the supplied data and control signals (the examiner interprets that the claims do not specify how many instructions are needed to make a set complete; the examiner interprets that if the required operation can be performed from just the one received instruction, then it is considered as a complete set of instruction; the single display instruction received by the graphics from the sub-routine process in response to the control signals correspond to a complete set of instruction; this instruction in the sub-routine is executed by the graphics subunit for rendering the concerned graphics primitive, and therefore it is considered as a complete set of instruction). Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to have the memory return a complete set of instructions as taught by Wang to the texture module of Rivard because this instruction set is needed to render the concerned graphics primitive (col. 5 lines 43-45 and lines 63-67)."

First, appellant respectfully asserts that in response to appellant's above arguments, the Examiner has merely reiterated the same arguments made in the Office Action mailed 09/20/2007.

Again, appellant agrees with the Examiner's argument that "Rivard does not explicitly teach a complete instruction set is received in response to the instruction request." Further, appellant respectfully disagrees with the Examiner's arguments with respect to Wang and asserts that the excerpts from Wang relied upon by the Examiner merely disclose that "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" where "[t]he display list generally contains instructions regarding the rendering of several graphic primitives, e.g., individual

points, lines, polygons, fills, BIT BLTs (bit block transfers), textures, etc." (Col. 5, lines 40-46 – emphasis added). Further, the excerpts disclose that "[m]any of the polygon display instructions include texture data to be displayed within the polygon" (Col. 5, lines 46-48).

However, appellant asserts that Wang's disclosure of "a 3D graphics subunit 109 for executing a series of display instructions" (emphasis added), clearly fails to support the Examiner's allegation that "if the required operation can be performed from just the one received instruction, then it is considered as a complete set of instruction" (emphasis added). Clearly, a series of display instructions fails to support "just the one received instruction" (emphasis added), as alleged by the Examiner. Furthermore, Wang's disclosure of a series of display instructions simply fails to meet appellant's claimed "complete instruction set," much less disclose that "a complete instruction set is received in response to the instruction request," where the "instructions [are received] from the video memory" (emphasis added), in the context as claimed by appellant (see Claim 1 for context). Therefore, appellant asserts that it would not have been obvious to "have the memory return a complete set of instructions as taught by Wang to the texture module of Rivard because this instruction set is needed to render the concerned graphics primitive," as alleged by the Examiner.

Again, appellant respectfully asserts that at least the first and third elements of the *prima facie* case of obviousness have not been met, since it would be *unobvious* to combine the references, as noted above, and the prior art excerpts, as relied upon by the Examiner, fail to teach or suggest all of the claim limitations, as noted above.

        Group #6: Claims 19 and 20

In addition, with respect to Claim 19, the Examiner has relied on Col. 5, lines 43-67; and Col. 6, lines 1-47 from Wang to make a prior art showing of appellant's claimed technique "wherein a partial instruction set is received in response to the instruction request." Specifically, the Examiner has argued that "Wang... teaches a graphics subunit (texture module) of a graphics hardware system executing a series of display instructions (set of instructions) stored in computer memory" and that "[t]he graphics subunit receives display instructions including texture data based on the supplied data and control signals."

Appellant respectfully disagrees and asserts that Wang merely discloses that "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for <u>executing a series of display instructions found within a display list stored in computer memory</u>" where "[t]he <u>display list generally contains instructions</u> regarding the rendering of several graphic primitives, e.g., individual points, lines, polygons, fills, BIT BLTs (bit block transfers), textures, etc." (Col. 5, lines 40-46 – emphasis added). However, only generally disclosing the graphics subunit executing <u>a series of display instructions</u> found within <u>a display list stored in computer memory</u>, as in Wang, does not specifically teach that "<u>a partial instruction set</u> is received in response to the instruction request," where the "instructions [are received] <u>from the video memory</u>" (emphasis added), in the context as claimed by appellant (see Claim 1 for context).

In fact, appellant notes that the Examiner has relied on the same disclosure in Wang of a "series of display instructions" to meet both appellant's claimed "complete instruction set [that] is received in response to the instruction request" (see Claim 18) and "partial instruction set [that] is received in response to the instruction request" (see Claim 19), as claimed. Clearly, a generally disclosure of a series of instructions, as in Wang, simply cannot meet appellant's claimed "complete instruction set" (Claim 18) <u>and</u> "partial instruction set" (Claim 19), as claimed.

Furthermore, in the Office Action mailed 09/20/2007, Page 12, line 12 to Page 13, line 9, the Examiner has argued the following:

> "Regarding claims 19, Rivard does not explicitly teach a partial instruction set is received in response to the instruction request. However, Wang (col. 5 lines 43-67, col. 6 lines 1-47) teaches a graphics subunit (texture module) of a graphics hardware system executing a series of display instructions (set of instructions) stored in computer memory. The graphics subunit receives display instructions including texture data based on the supplied data and control signals (the examiner interprets that the claims do not specify how many instructions are needed to make a set complete or partial; the examiner interprets that if the required operation can be performed from just the one received instruction, then it is considered as a complete set of instruction; however, the rendering process might need other instructions for rendering other primitives, and in this case, the single received instruction is considered as a partial set of instruction; the single display instruction received by the graphics from the sub-routine process in response to the control signals correspond to a partial set of instruction; this instruction in the sub-routine is executed by the graphics subunit for rendering other primitives, and therefore overall

this single instruction is considered as a partial set of instruction). Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device (col. 5 lines 43-45 and lines 63-67)." (Page 12, line 12 to Page 13, line 9)

Appellant agrees with the Examiner's argument that "Rivard does not explicitly teach a partial instruction set is received in response to the instruction request." Further, appellant respectfully disagrees with the Examiner's arguments with respect to the Wang reference and asserts that the excerpts from Wang relied upon by the Examiner disclose that "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" where "[t]he display list generally contains instructions regarding the rendering of several graphic primitives, e.g., individual points, lines, polygons, fills, BIT BLTs (bit block transfers), textures, etc." (Col. 5, lines 40-46 – emphasis added). Further, the excerpts disclose that "[m]any of the polygon display instructions include texture data to be displayed within the polygon" (Col. 5, lines 46-48).

However, appellant asserts that Wang's disclosure of "a 3D graphics subunit 109 for executing a series of display instructions" (emphasis added), clearly fails to support the Examiner's allegation that "if the required operation can be performed from just the one received instruction, then it is considered as... a partial set of instruction" (emphasis added). Clearly, a series of display instructions fails to support "just the one received instruction," as alleged by the Examiner. Furthermore, Wang's disclosure of a series of display instructions simply fails to meet appellant's claimed "partial instruction set," much less disclose that "a partial instruction set is received in response to the instruction request," where the "instructions [are received] from the video memory" (emphasis added), in the context as claimed by appellant (see Claim 1 for context). Therefore, appellant asserts that it would not have been obvious to "have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device," as alleged by the Examiner.

In the Examiner's Answer mailed 06/26/2008, on Pages 59-60, the Examiner has argued the following:

"However, the examiner interprets that Rivard does not explicitly teach a partial instruction set is received in response to the instruction request. However, Wang (col. 5 lines 43-67, col. 6 lines 1-47) teaches a graphics subunit (texture module) of a graphics hardware system executing a series of display instructions (set of instructions) stored in computer memory. The graphics subunit receives display instructions including texture data based on the supplied data and control signals (the examiner interprets that the claims do not specify how many instructions are needed to make a set complete or partial; the examiner interprets that if the required operation can be performed from just the one received instruction, then it is considered as a complete set of instruction; however, the rendering process might need other instructions for rendering other primitives, and in this case, the single received instruction is considered as a partial set of instruction; the single display instruction received by the graphics from the subroutine process in response to the control signals correspond to a partial set of instruction; this instruction in the sub-routine is executed by the graphics subunit for rendering the concerned graphics primitive, but other instructions are needed for rendering other primitives, and therefore overall this single instruction is considered as a partial set of instruction). Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device (col. 5 lines 43-45 and lines 63-67)."

Appellant agrees with the Examiner's argument that "Rivard does not explicitly teach a partial instruction set is received in response to the instruction request." Further, appellant respectfully disagrees with the Examiner's arguments with respect to the Wang reference and asserts that the excerpts from Wang relied upon by the Examiner disclose that "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" where "[t]he display list generally contains instructions regarding the rendering of several graphic primitives, e.g., individual points, lines, polygons, fills, BIT BLTs (bit block transfers), textures, etc." (Col. 5, lines 40-46 – emphasis added). Further, the excerpts disclose that "[m]any of the polygon display instructions include texture data to be displayed within the polygon" (Col. 5, lines 46-48).

However, appellant asserts that Wang's disclosure of "a 3D graphics subunit 109 for executing a series of display instructions" (emphasis added), clearly fails to support the Examiner's allegation that "if the required operation can be performed from just the one received instruction, then it is considered as… a partial set of instruction" (emphasis added). Clearly, a series of display instructions fails to support "just the one received instruction," as alleged by the Examiner. In addition, Wang's disclosure of executing a series of display instructions found

within a display list, which generally contains **instructions** regarding the rendering of several graphic primitives, simply fails to support the Examiner's allegation that "this instruction in the sub-routine is executed by the graphics subunit for rendering the concerned graphics primitive, but other instructions are needed for rendering other primitives, and therefore overall this single instruction is considered as a partial set of instruction" (emphasis added), as alleged by the Examiner. Furthermore, Wang's disclosure of a series of display instructions simply fails to meet appellant's claimed "partial instruction set," much less disclose that "a partial instruction set is received in response to the instruction request," where the "instructions [are received] from the video memory" (emphasis added), in the context as claimed by appellant (see Claim 1 for context). Therefore, appellant asserts that it would not have been obvious to "have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device," as alleged by the Examiner.

Again, appellant respectfully asserts that at least the first and third elements of the *prima facie* case of obviousness have not been met, since it would be *unobvious* to combine the references, as noted above, and the prior art excerpts, as relied upon by the Examiner, fail to teach or suggest all of the claim limitations, as noted above.

Issue #2:

The Examiner has rejected Claims 13-17, 22, 23, and 29 under 35 U.S.C. 103(a) as being unpatentable over Rivard (U.S. Patent No. 5,987,567), in view of Wang (U.S. Patent No. 5,831,640), and further in view of Applicant Admitted Prior Art (AAPA).

Group #1: Claims 13-17, 22 and 23

Appellant respectfully asserts that such claims are not met by the prior art for the reasons argued with respect to Issue #1, Group #1.

Group #2: Claim 29

Furthermore, with respect to independent Claim 29, the Examiner has relied on Figures 6 and 10; Col. 4, lines 46-57; Col. 6, lines 45-67; and Col. 7, lines 1-10 from Rivard, along with Fig. 3; and Page 5, lines 24-31 from AAPA to make a prior art showing of appellant's claimed "sending an instruction request to video memory, where a texture module coupled to the shader module sends the instruction request to the video memory."

Appellant respectfully asserts that the figures relied on by the Examiner only generally illustrate a computer system and a block diagram detailing graphics accelerator system showing the pipeline latency elements 1025. In addition, in Col. 7, lines 4-7, Rivard teaches that "graphics accelerator system 635 includes pipeline latency elements 1025 to coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030" (emphasis added). Furthermore, Rivard teaches that "[m]emory request generator 1020 generates memory requests for all misses (up to four for bi-linear sampling and up to eight for tri-linear sampling), and forwards the requests..." (Col. 6, lines 50 – 52 emphasis added). However, disclosing that memory requests are generated for all misses, as in Rivard, fails to suggest "sending an instruction request to video memory, where a texture module coupled to the shader module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

In addition, appellant respectfully asserts that Fig. 3 from AAPA as relied upon by the Examiner merely discloses that "the texels resulting from one texture lookup can influence the location of the texels in a subsequent texture lookup" (Page 4, lines 30-31 – emphasis added). However, the mere disclosure of texels resulting from a texture lookup fail to even suggest "sending an instruction request to video memory, where a texture module coupled to the shader module sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Clearly, retrieving texels from a texture lookup fails to meet "an instruction request," in the manner as claimed by appellant.

In the Office Action mailed 03/20/2007, the Examiner has argued that "Rivard teaches sending an instruction request to video memory, where the texture module sends the instruction to the video memory." Appellant disagrees with the Examiner's arguments and asserts that, for substantially the same reasons as those argued above with respect to at least some of the

independent claims, Rivard fails to even suggest "sending an instruction request," as appellant claims. For example, appellant respectfully points out that Rivard merely teaches that "the memory request generator is coupled between the cache tag block and the cache data store for performing a memory request before an address and instruction information associated with the needed texels reach the cache data store" (Col. 10, lines 48-52 - emphasis added). Clearly, performing a memory request before the address and instruction information reach the cache data store, as in Rivard (see excerpts above), fails to teach "sending an instruction request" (emphasis added), as claimed by appellant.

Furthermore, with respect to Claim 29 and the Examiner's reliance upon the rejection of Claim 1, the Examiner has argued that "the definition of instruction provided by dictionary.com, which defines instructions as a command given to a computer to carry out a particular operation and can contain data to be used in the operation; here DRAM sends memory data based on the memory requests/read requests from the texture module." Further, the Examiner has argued that "[m]emory requests/read requests act as an instruction to DRAM, which performs a particular function based on the request."

Appellant respectfully disagrees with the Examiner's arguments and asserts that the dictionary.com reference provided by the Examiner merely defines an instruction as "a command given to a computer to carry out a particular operation." Appellant respectfully asserts that Rivard merely discloses that the "cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" which "generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-36, and lines 48-53 – emphasis added). However, the disclosure of the memory request generator generating a memory request and forwarding the request to DRAM for information retrieval, as in Rivard, simply fails to specifically suggest "sending an instruction request to video memory, where a texture module coupled to the shader module sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Clearly, generating and forwarding a memory request to DRAM, as in Rivard, fails to suggest "sending an instruction request" (emphasis added), in the manner as claimed by appellant.

In the Office Acton mailed 09/20/2007, on Page 2, line 19 to Page 3, line 19; and Page 4, lines 3-7, the Examiner has argued the following:

"However, the examiner interprets that Rivard (Fig. 6, Fig. 10, col. 4 lines 46-57, col. 6 lines 45-67, col. 7 lines 1-10) teaches sending an instruction request to video memory, where a texture module in a graphics pipeline sends the instruction request to the video memory (a texture mapping stage sends information to texel cache system; the texture mapping stage and the texel cache system together are considered as texture module, so the information is passed between the components of the texture module; texel cache system comprising the memory request generator, pipeline latency elements, and cache data store and memory data resolver forwards the memory requests/read requests to DRAM for information retrieval. [T]he examiner interprets that Rivard discloses sending an instruction request to video memory by generating memory request for all misses, and forwards the requests to DRAM for information retrieval (Fig. 6, Fig. 10, col. 4 lines 46-57, col. 6 lines 45-67, col. 7 lines 1-10). It is interpreted that an instruction request could be considered as merely a request because it is basically a request that requires DRAM to perform necessary operation. The examiner also interprets that instructions stored in DRAM that are requested are basically stored as data. It is further interpreted that memory requests/read requests act as an instruction to DRAM based on which DRAM performs a particular function. Memory requests/read requests for all misses that are forwarded to DRAM are instruction requests based on which DRAM sends back information; the examiner relies on the definition of instruction provided by dictionary.com, which defines instructions as a command given to a computer to carry out a particular operation and can contain data to be used in the operation; here DRAM sends memory based on the memory requests/read requests from the texture module." (Page 2, line 19 to Page 3, line 19)

"In response to [appellant's] arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986)." (Page 4, lines 3-7)

First, appellant respectfully asserts that the Examiner's arguments on Page 4, line 8 to Page 5, line 8 in the Office Action mailed 09/20/2007 are substantially similar to the arguments made by the Examiner on Page 2, line 19 to Page 3, line 19 excerpted above. Second, appellant has clearly responded to such arguments above regarding how the excerpts repeatedly relied upon by the Examiner fail to meet or suggest appellant's claimed "sending an instruction request to video memory, where a texture module coupled to the shader module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Further, appellant respectfully disagrees with the Examiner's arguments and asserts that the figures and excerpts from Rivard relied upon by the Examiner merely disclose that "the texture mode indicates a 16-bit per pixel texture lookup" in addition to "satisfy[ing] two lookup requests with a single read request" (Col. 6, lines 56-61 – emphasis added). Additionally, the excerpts from Rivard teach that "pipeline latency elements 1025... coordinate arrival of the memory data and of the associated instructions" (Col. 7, lines 3-7 – emphasis added) such that "the memory request generator... perform[s] a memory request before an address and instruction information associated with the needed texels reach the cache data store" (Col. 10, lines 48-52 – emphasis added). Clearly, the texture mode indicating a texture lookup, in addition to the memory request generator performing a memory request, as in Rivard, simply fails to support the Examiner's allegation that "a texture module... sends the instruction request to the video memory" (emphasis added), and especially does not teach "sending an instruction request to video memory, where a texture module coupled to the shader module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

In addition, Rivard's disclosure that the "cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020," which "generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 48-53 – emphasis added). However, the disclosure of the memory request generator generating a memory request and forwarding the request to DRAM for information retrieval, as in Rivard, fails to support the Examiner's allegation "that Rivard discloses sending an instruction request to video memory by generating memory request for all misses, and forwards the requests to DRAM for information retrieval" (emphasis added). Clearly, generating a memory request and forwarding the request to DRAM for information retrieval, as in Rivard, simply fails to specifically teach "sending an instruction request to video memory, where a texture module coupled to the shader module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Still yet, appellant respectfully asserts that the excerpts from Rivard relied upon by the Examiner disclose that the "[g]raphics application program 670 transfers graphical information from data storage 625 into DRAM 655 for local storage" (Col. 4, lines 46-48 – emphasis added). Additionally, the excerpts disclose that "[i]f a hit occurs, then cache tag blocks 1010 and 1015

forward a cache read address through memory request generator 1020 and through pipeline latency elements 1025 to cache data store 1030" such that "[c]ache data store 1030 responsively outputs on lines 649 the texture values cached in the read location" (Col. 6, lines 42-47 – emphasis added). Clearly, Rivard is disclosing the transfer of graphical information into DRAM and outputting cached texture values, which simply fails to support the Examiner's allegation that "instructions stored in DRAM that are requested are basically stored as data" (emphasis added). Therefore, the disclosure of transferring graphical information into DRAM and outputting cached texture values, as in Rivard, simply fails to suggest "sending an instruction request to video memory, where a texture module coupled to the shader module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Furthermore, appellant again asserts that the dictionary.com reference provided by the Examiner merely defines an instruction as "a command given to a computer to carry out a particular operation." In addition, the excerpts from Rivard disclose that the "memory request generator 1020 generates memory requests for all misses… and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 50-53 – emphasis added). However, generating and forwarding memory requests to DRAM for information retrieval, as in Rivard, fails to support the Examiner's allegation that "[m]emory requests/read requests for all misses that are forwarded to DRAM are instruction requests based on which DRAM sends back information" (emphasis added). Clearly, generating and forwarding memory requests to DRAM for information retrieval, as in Rivard, simply fails to specifically teach "sending an instruction request to video memory, where a texture module coupled to the shader module sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Appellant emphasizes that the memory request of Rivard fails to meet appellant's claimed "instruction request," as claimed by appellant.

Still yet, appellant respectfully disagrees with the Examiner's argument that "it would have been obvious to a person of ordinary skill in the art at the time of the invention to incorporate the shading module of AAPA into the texture module of Rivard and Wang because combination of shading module and texture module would enable a shading function of the graphics pipeline." Clearly, the excerpts from AAPA, Rivard, and Wang relied upon by the Examiner fail to suggest "sending an instruction request to video memory, where a texture module coupled to the shader

module sends <u>the instruction request</u> to the video memory" (emphasis added), as claimed by appellant.

Furthermore, in the Office Action mailed 09/20/2007, in paragraph 15 on page 10, the Examiner has argued the following:

"In response to [appellant's] arguments against references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986).

In this case, the examiner interprets that Rivard, in view of Wang teaches to send an instruction request to memory utilizing a texture module (see refer to the arguments above for details).

Although the combination of Rivard and Wang teach a method and system for retrieving instructions from memory utilizing a texture module in a graphics pipeline, they do not explicitly teach controlling the texture module utilizing a shader module coupled thereto. However, AAPA teaches controlling the texture module utilizing a shader module coupled thereto (Fig. 3; shader module is also coupled to the rasterizer module). Therefore, it would have been obvious to a person of ordinary skill in the art at the time of the invention to incorporate the shading module of AAPA into the texture module of Rivard and Wang because combination of shading module and texture module would enable a shading function of the graphics pipeline."

Appellant respectfully disagrees.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 61 and 67, the Examiner has argued the following:

"In response to [appellant's] arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986)."

Appellant respectfully disagrees, and asserts that Wang teaches that "the <u>graphics subunit 109</u> including <u>a texture engine 10</u>, a polygon engine 12 and a pixel pipeline 16" (Col. 6, lines 3-5 – emphasis added) and that Fig. 2 illustrates that <u>the texture engine 10</u> further contains <u>a texture map data access circuit 200</u>, where the "[t]exture map data retrieval (TDA) circuit 200 performs <u>texture map data</u> retrieval processes" (Col. 6, lines 23-25 – emphasis added). Further, Wang

teaches that "[t]he host computer system 112 provides data and control signals via bus 100 to a graphics hardware unit or system, e.g., 'graphics card' 108," where "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" (Col. 5, lines 38-43 – emphasis added). In addition, Rivard teaches, as illustrated in Fig. 10, that the texel cache system 650 "includes cache tags 1010, cache tags 1015, a memory request generator 1020, pipeline latency elements 1025 and cache data store and memory data resolver 1030" (Col. 6, lines 22-26), and that "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57 – emphasis added).

However, disclosing that the graphics subunit 109 includes a texture engine 10 that contains a texture map data access circuit 200 that performs a texture map data retrieval processes, in addition to disclosing that the host computer system 112 provides data and control signals via bus 100 to a graphics hardware unit that contains a graphics subunit 109 that executes a series of display instructions found within a display list stored in computer memory, as in Wang, in addition to disclosing that texture mapping block 645 sends information via bus 647 to texel cache system 650 that sends information via bus 649 back to texture mapping block 645, where the texel cache system 650 includes cache, a memory request generator, pipeline latency elements, and cache data store and memory data resolver, as in Rivard, simply fails to suggest "sending an instruction request to video memory, where a texture module coupled to the shader module sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Clearly, Wang's disclosure that the graphics subunit 109 executes a series of display instructions found within a display list stored in computer memory, in addition to Rivard's disclosure that the texture mapping block 645 sends information via bus 647 to texel cache system 650 that further sends information via bus 649 back to texture mapping block 645, simply fails to suggest that "a texture module... sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 61-62, 66, 67-68 and 72-73 the Examiner has argued the following:

"However, the examiner interprets that Rivard (Fig. 6, Fig. 10, col. 4 lines 46-57, col. 6 lines 45-67, col. 7 lines 1-10) teaches sending an instruction request to video memory, where a texture module in a graphics pipeline sends the instruction request to the video memory (a texture mapping stage sends information to texel cache system; the texture mapping stage and the texel cache system together are considered as texture module, so the information is passed between the components of the texture module; texel cache system comprising the memory request generator, pipeline latency elements, and cache data store and memory data resolver forwards the memory requests/read requests to DRAM for information retrieval; the examiner interprets that Rivard discloses sending an instruction request to video memory by generating memory request for all misses, and forwards the requests to DRAM for information retrieval). It is interpreted that an instruction request could be considered as merely a request because it is basically a request that requires DRAM to perform necessary operation. The examiner also interprets that a DRAM can store instructions as data, and that the memory request retrieves such data from DRAM. It is further interpreted that when there is a miss, a memory requests/read requests is send to DRAM, which acts as an instruction to DRAM based on which DRAM performs a particular function of returning data for further processing. Therefore, it is reasonable to interpret that memory requests/read requests for all misses that are forwarded to DRAM are instruction requests based on which DRAM sends back information to the graphics pipeline for further processing (it should be noted that the examiner relies on the definition of instruction provided by dictionary.com, which defines instructions as a command given to a computer to carry out a particular operation and can contain data to be used in the operation; in this case, when the memory/read request is send to DRAM, DRAM sends memory data based on the request to the texture module of the graphics pipeline for further processing)."

"The above argument is similar in scope to the arguments discussed above. Please refer to the statements presented above in regards to the reasons provided for above arguments."

Appellant respectfully disagrees and asserts that Rivard teaches that "[g]raphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650," where "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 49-57 – emphasis added).

However, a graphics accelerator system 635 including graphics pipeline stages 640 that includes a texture mapping stage 645, where the texture mapping stage 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645, as in Rivard, simply fails to support the Examiner's assertion that

"the texture mapping stage and the texel cache system together are considered as texture module," as alleged by the Examiner. Furthermore, Rivard's teachings that the graphics pipeline stages 640 include a texture mapping stage 645, where the texture mapping stage 645 sends information via bus 647 to texel cache system 650, simply fails to suggest that "sending an instruction request to video memory, where a texture module coupled to the shader module sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Clearly, teaching that the texture mapping block 645, included in the graphics pipeline stages 640, sends information via bus 647 to the separate texel cache system 650, which is **not included** in the graphics pipeline stages 640, as in Rivard, simply fails to even suggest that "a texture module coupled to the shader module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Additionally, appellant respectfully asserts that Rivard teaches that, as illustrated in Fig. 10, the "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015," where the "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," and that "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that "generates memory requests for all misses..., and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-53 – emphasis added).

Therefore, Rivard teaches that the texel sample address computation block 1005 of the texture mapping block 645 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015, where if the cache tag blocks determine that the requested texel values are not stored in cache data store 1030, then the cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020 that generates memory requests for all misses and forwards the requests to DRAM 655 for information retrieval. Clearly, generating a memory request when the cache tag blocks determine that the requested texel values are not stored in cache data store, and forwarding the memory request to DRAM for information retrieval, as in Rivard, simply fails to support the Examiner's allegation that "an instruction request could be considered as merely a request because it is basically a request that requires DRAM to perform necessary operation" (emphasis

added), as alleged by the Examiner, and further fails to suggest appellant's claimed "instruction request," as claimed by appellant. Clearly, generating a memory request when the requested texel values are not stored in cache data store, as in Rivard, simply fails to even suggest appellant's claimed "instruction request," as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 62-63, 68-69 and 73-74, the Examiner has argued the following:

> "The examiner further interprets that Rivard does not explicitly teach texture mapping stage and texel cache system together functions as a texture module. However, it would have been obvious to one of ordinary skill in art at the time of present invention to combine texture mapping stage and texel cache system of Rivard to work together as a texture module (it should be noted that the examiner interprets texture mapping stage and texel cache system combined together will perform functions similar to the claimed texture module, and therefore it would be obvious to modify the Rivard reference). The unity of diversity of parts would depend more upon the choice of the manufacturer, and the convenience and availability of the machines and tools necessary to construct the texture module, than on any inventive concept. One of ordinary skill in art, furthermore, would have expected [appellant's] invention to perform equally well with Rivard's reference that teaches to send and receive information/instruction from the texture mapping stage and texel cache system to the DRAM because using this components together will also result in sending and receiving instructions to and from DRAM. Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to modify Rivard to obtain the invention as specified in the claim."

First, appellant respectfully agrees with the Examiner's assertion that "Rivard does not explicitly teach texture mapping stage and texel cache system together functions as a texture module," since, as argued hereinabove, Rivard teaches that the texture mapping stage 645, included in the graphics pipeline stages 640, sends information via bus 647 to the separate texel cache system 650, which is **not included** in the graphics pipeline stages 640.

Second, appellant respectfully disagrees with the Examiner's allegation that "it would have been obvious to one of ordinary skill in art at the time of present invention to combine texture mapping stage and texel cache system of Rivard to work together as a texture module," as alleged by the Examiner. Appellant asserts that Rivard teaches that "[a] system for caching texel information in a cache data store, for use in a graphics rendering system," where the "system includes a texel memory storing texel information, ...a first cache data storage..., a second cache data storage..., cache tag blocks..., and a memory request generator..." (Abstract – emphasis

added). Further, Rivard teaches, as illustrated in Fig. 10, that the texel cache system 650 "includes cache tags 1010, cache tags 1015, a memory request generator 1020, pipeline latency elements 1025 and cache data store and memory data resolver 1030" (Col. 6, lines 22-26), and that "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57 – emphasis added).

Therefore, Rivard is clearly teaching that the use of bus 647 for sending information from the texture mapping stage 645, included in the graphics pipeline stages 640, to the separate texel cache system 650, and the use of bus 649 for sending information from the texel cache system 650 back to texture mapping stage 645, which fails to suggest that it would be obvious "to combine texture mapping stage and texel cache system of Rivard to work together as a texture module," as alleged by the Examiner, and further fails to suggest that "a texture module coupled to the shader module sends the instruction request to the video memory" (emphasis added), as claimed by appellant. Furthermore, appellant thus formally requests a specific showing of the subject matter in ALL of the claims in any future action. Note excerpt from MPEP below.

> "If the [appellant] traverses such an [Official Notice] assertion the examiner should cite a reference in support of his or her position." See MPEP 2144.03.

Third, appellant respectfully asserts that Rivard teaches that, as illustrated in Fig. 10, the "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015," where the "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," and that "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that "generates memory requests for all misses..., and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-53 – emphasis added). Additionally, Figs. 6 and 10 clearly illustrate that the texel sample address computation block 1005 of Fig. 10 is a part of the texture mapping stage 645 that sends information via bus 647 to the cache tag blocks 1010 and 1015 of the texel cache system 650.

Therefore, Rivard teaches that the texel sample address computation block 1005 of the texture mapping stage 645 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015 via bus 647 (as illustrated in Fig. 10), where if the cache tag blocks determine that the requested texel values are not stored in cache data store 1030, then the cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020 that generates memory requests for all misses. However, the texture mapping stage 645 forwarding sample points to the cache tag blocks of the texel cache system 650, where the cache tag blocks forward a cache write address to the memory request generator that generates the memory request, as in Rivard, fails to support that Examiner's allegation that the Rivard "reference that teaches to send and receive information/instruction from the texture mapping stage and texel cache system to the DRAM because using this components together will also result in sending and receiving instructions to and from DRAM" (emphasis added), as alleged by the Examiner, and further fails to suggest that "a texture module coupled to the shader module sends the instruction request to the video memory" (emphasis added), as claimed by appellant.

Furthermore, Rivard teaches that "[i]f the interface to DRAM 655 is 32-bits wide, the texture mode indicates a 16-bit per pixel texture lookup and the data is conveniently aligned in the texture map, then it is possible to satisfy two lookup requests with a single read request" (Col. 6, lines 56-61 – emphasis added). Clearly, the texture mode indicating a 16-bit per pixel texture lookup, as in Rivard, simply fails to support the Examiner's allegation of "sending and receiving instructions to and from DRAM" (emphasis added), as alleged by the Examiner.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Page 66, in addition to the paragraphs on Page 77, the Examiner has argued the following:

> "Although the combination of Rivard and Wang teach a method and system for retrieving instructions from memory utilizing a texture module in a graphics pipeline, they do not explicitly teach controlling the texture module utilizing a shader module coupled thereto. However, AAPA teaches controlling the texture module utilizing a shader module coupled thereto (Fig.3; shader module is also coupled to the rasterizer module). Therefore, it would have been obvious to a person of ordinary skill in the art at the time of the invention to incorporate the shading module of AAPA into the texture module of Rivard and Wang because combination of shading module and texture module would enable a shading function to the graphic pipeline."

"The above argument is similar in scope to the arguments discussed above. Please refer to the statements presented above in regards to the reasons provided for above arguments."

Appellant respectfully disagrees.

In addition, with respect to Claim 29, the Examiner has relied on Figures 6 and 10; Col. 4, lines 46-57; Col. 6, lines 45-67; and Col. 7, lines 1-10 from Rivard, along with Fig. 3; and Page 5, lines 24-31 from AAPA to make a prior art showing of appellant's claimed "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module."

Specifically, the Examiner has argued that "Rivard... teaches that DRAM returns memory data to cache data store and memory data resolver component of texel cache system, which further sends this information to the texture mapping stage." Further, the Examiner has argued that "the texture mapping stage and the texel cache system together are considered as texture module, so the information is passed between the components of the texture module." In addition, the Examiner has argued that "[t]he texture mapping stage as shown in Fig. 6 also receives data (instructions) from the rasterizer module of the pipeline, and thus the instructions received from the DRAM via the texel cache system are considered to be the additional instructions."

Appellant respectfully disagrees with the Examiner's arguments and asserts that Rivard merely discloses that "DRAM 655 returns the memory data on bus 660 to cache data store and memory data resolver 1030, which stores the memory data at the write address" (Col. 6, lines 53-56 – emphasis added). However, merely storing memory data at the write address, as in Rivard, simply fails to suggest "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant. Clearly, storing memory data, as in Rivard, fails to suggest "receiving additional instructions," in the manner as claimed by appellant.

Second, in response to the Examiner's argument that "the texture mapping stage and the texel cache system together are considered as texture module," appellant respectfully disagrees and asserts that Rivard merely teaches that "[g]raphics accelerator system 635 includes graphics

pipeline stages 640 including <u>a texture mapping stage 645</u> for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the <u>texel information</u> in a texel cache system 650" where the "[t]exture mapping block 645 <u>sends information</u> via bus 647 <u>to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645</u>" (Col. 4, lines 49-57 – emphasis added). Clearly, the texture mapping stage and separate texture cache system, as in Rivard, fails to suggest "<u>receiving additional instructions</u> from the video memory in response to the <u>instruction request utilizing the texture module</u>" (emphasis added), as claimed by appellant.

Third, in response to the Examiner's argument that "Fig. 6 also receives data (instructions) from the rasterizer module of the pipeline, and thus the instructions received from the DRAM via the texel cache system are considered to be the additional instructions," appellant respectfully disagrees and asserts that Rivard merely teaches that "the <u>memory request generator</u> is coupled between the cache tag block and the cache data store for performing <u>a memory request</u> <u>before</u> an address and instruction information associated with the needed texels reach the cache data store" (emphasis added). Clearly, coordinating the arrival of "<u>memory data</u>", where "the data is conveniently aligned in the <u>texture map</u>" (emphasis added), as in Rivard (see excerpts above), fails to teach "receiving <u>additional instructions</u>" (emphasis added), as claimed by appellant. In addition, appellant respectfully asserts that simply nowhere in the description of Figure 6 is there any disclosure that "Fig. 6 also receives data (instructions) from the rasterizer module of the pipeline," as noted by the Examiner. Thus, Rivard simply fails to meet appellant's claimed "receiving additional instructions… utilizing the texture module," as claimed.

Furthermore, in the Office Action mailed 09/20/2007, on Page 7, lines 3-7; Page 5, lines 9-17; Page 6, line 10 to Page 7, line 4; and Page 8, lines 4-8, the Examiner has argued the following:

> "In response to [appellant's] arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986)." (Page 7, lines 3-7)

> "Rivard further teaches receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline (it is interpreted that instructions stored in DRAM that are requested are basically stored as

data, and Rivard teaches to send back data in some form in response to the memory request generated as a result of a miss; DRAM returns memory data to cache data store and memory data resolver component of texel cache system, which further sends this information to the texture mapping stage; the texture mapping stage and the texel cache system together are considered as texture module, so the information/memory data is passed between the components of the texture module)." (Page 5, lines 9-17)

"The examiner further interprets that Rivard also teaches the arrival of memory data and associated instructions at the cache data store and memory data resolver component of the text cache system is coordinated by pipeline latency elements (col. 7 lines 4-7; memory data has associated instructions are returned by DRAM, though the data has it's associated instructions). Therefore, the examiner brings in the Wang reference that suggests a graphics subunit (texture module) in a graphics hardware system that supplies data and control signals to local frame buffer memory for executing a series of display instructions (Fig. 1, col. 5 lines 38-67; the computer memory includes the local frame buffer memory, which corresponds to the DRAM; based on the control signals send by the graphics hardware system, the frame buffer returns the polygon display instructions that includes the texture data, so that the graphics subunit executes the display instructions using this data). Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device (col. 5 lines 43-45 and lines 63-67)." (Page 6, line 10 to Page 7, line 4)

"However, the examiner interprets that Rivard teaches to send a request to DRAM via a texture module, and based on this request, DRAM sends back data and it's associated information to the texture module (see the arguments above for details). Nonetheless, Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM, though the data has it's associated functions." (Page 8, lines 4-8)

First, appellant notes that the Examiner's arguments on Page 7, lines 5-21; and Page 8, lines 9-20 in the Office Action mailed 09/20/2007 are substantially similar to the arguments made by the Examiner on Page 6, line 10 to Page 7, line 4 excerpted above. Further, on Page 7, line 21 the Examiner has additionally citied "*In re Lockhart*, 90 USPQ (CCPA 1951)." Second, appellant has clearly responded to such arguments above regarding how the excerpts repeatedly relied upon by the Examiner fail to meet or suggest appellant's claimed "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant.

Further, appellant respectfully asserts that the excerpts from Rivard relied upon by the Examiner merely teach that "[m]emory request generator 1020 generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" such that "DRAM 655 returns

the memory data on bus 660 to cache data store and memory data resolver 1030" (Col. 6, lines 50-55 – emphasis added). However, Rivard's disclosure that memory requests are forwarded to DRAM for information retrieval, where the DRAM returns the memory data, fails to support the Examiner's allegation that "Rivard further teaches receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline" (emphasis added). Clearly, forwarding memory requests to DRAM for information retrieval, as in Rivard, simply fails to specifically teach "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant.

In addition, appellant respectfully asserts that the excerpts from Rivard relied upon by the Examiner teach that "[b]ecause memory request generator 1020 is between cache tag blocks 1010, 1015 and cache data store 1030, generator 1020 can perform DRAM 655 memory requests before the address and instruction information reach cache data store and memory data resolver 1030" (Col. 6, lines 62-66 – emphasis added). Furthermore, Wang teaches "a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" and that "[m]any of the polygon display instructions include texture data to be displayed within the polygon" (Col. 5, lines 40-48 – emphasis added). However, the mere disclosure of memory request generator 1020 performing DRAM memory requests before the instruction information reaches the cache data store, as in Rivard, in addition to the disclosure of executing display instructions that may include texture data, as in Wang, simply fails to suggest "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant. Clearly, performing memory requests before instruction information reaches the cache data store, as in Rivard, in addition to the disclosure of executing display instructions that may include texture data, as in Wang, simply fails to even suggest an "instruction request," much less "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant.

Further, appellant respectfully asserts that Rivard's disclosure of a "graphics accelerator system 635 [that] includes pipeline latency elements 1025 to coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030" (Col. 7, lines

4-7) clearly *teaches away* from the Examiner's allegation that it would be obvious to "have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device," as noted by the Examiner, since in Rivard the "pipeline latency elements... coordinate arrival of the memory data and... the associated instructions" (emphasis added). Clearly, as argued above, it would not be obvious from the teachings of Rivard and Wang to "receiv[e] additional instructions from the video memory in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant.

Still yet, appellant agrees with the Examiner's statement that "Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM." For example, appellant asserts that the figures and excerpts from Rivard relied upon by the Examiner teach that texel cache system 650 "includes cache tags 1010, cache tags 1015, a memory request generator 1020, pipeline latency elements 1025 and cache data store and memory data resolver 1030" (see Figure 10; Col. 6, lines 22-26 – emphasis added) where the "[m]emory request generator 1020 generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 50-53 – emphasis added). Clearly, a texel cache system 650 including a memory request generator 1020 that generates memory requests that are forwarded to DRAM 655 for information retrieval, as in Rivard, simply fails to suggest "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant.

Furthermore, in the Office Action mailed 09/20/2007, on Page 5, line 18 to Page 6, line 9, the Examiner has argued the following:

> "Although Rivard discloses the limitations as stated above, Rivard does not explicitly teach to combine texture mapping stage and texel cache system to form a texture module. However, it would have been obvious to one of ordinary skill in art at the time of present invention to combine texture mapping stage and texel cache system of Rivard to work together as a texture module. The unity of diversity of parts would depend more upon the choice of the manufacturer, and the convenience and availability of the machines and tools necessary to construct the texture module, than on any inventive concept. One of ordinary skill in art, furthermore, would have expected [appellant's] invention to perform equally well with Rivard's reference that teaches to send and receive information/instruction from the texture mapping stage and texel cache system to the DRAM because using this components together will also result in sending

and receiving instructions to and from DRAM. Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to modify Rivard to obtain the invention as specified in the claim." (Page 5, line 18 to Page 6, line 9)

First, appellant respectfully asserts that the Examiner's arguments on Page 7, lines 8-21 of the Office Action mailed 09/20/2007 are substantially similar to the arguments made by the Examiner on Page 5, line 18 to Page 6, line 9 excerpted above. Further, on Page 7, line 21 the Examiner has additionally citied "*In re Lockhart*, 90 USPQ (CCPA 1951)."

Further, appellant respectfully disagrees with the Examiner's arguments and asserts that it would not be obvious to "combine texture mapping stage and texel cache system of Rivard to work together as a texture module," as alleged by the Examiner. For example, Rivard teaches a "[g]raphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650" such that "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 49-57 – emphasis added). Clearly, Rivard teaches and suggests the use of graphic pipeline stages, such as a texture mapping block 645 and texel cache system 650, which simply fails to support the Examiner's allegation that it would be obvious to "combine texture mapping stage and texel cache system of Rivard to work together as a texture module." Therefore, Rivard's teachings of a separate texture mapping block 645 and texel cache system 650 simply fails to suggest "receiving additional instructions... in response to the instruction request utilizing the texture module" (emphasis added), as claimed.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 63-64, 69-70 and 74, the Examiner has argued the following:

> "Rivard further teaches receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline (the examiner interprets that a DRAM can store instructions as data, and that the memory request retrieves such data from DRAM; Rivard teaches to send back data in some form in response to the memory request generated as a result of a miss; DRAM returns memory data to cache data store and memory data resolver component of texel cache system, which further sends this information to the texture mapping stage; the texture

mapping stage and the texel cache system together are considered as texture module, so the information/memory data is passed between the components of the texture module)."

Appellant respectfully disagrees and asserts that Rivard teaches that, as illustrated in Fig. 10, the "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015," where the "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," and that "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that "generates memory requests for all misses..., and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-53 – emphasis added). Additionally, Figs. 6 and 10 clearly illustrate that the texel sample address computation block 1005 of Fig. 10 is a part of the texture mapping stage 645 that sends information via bus 647 to the cache tag blocks 1010 and 1015 of the texel cache system 650.

However, teaching that the texel sample address computation block 1005 of the texture mapping stage 645 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015 via bus 647 (as illustrated in Fig. 10), where if the cache tag blocks of the texel cache system 650 determine that the requested texel values are not stored in cache data store 1030, then the cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020 of the texel cache system 650 that generates memory requests for all misses and forwards the requests to DRAM 655 for information retrieval, as in Rivard, simply fails to suggest "receiving additional instructions... in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant. Clearly, the cache tag blocks of the texel cache system 650 determining that the requested texel values are not stored in cache data store 1030, in addition to the memory request generator 1020 of the texel cache system 650 generating and forwarding memory requests to DRAM 655 for information retrieval, as in Rivard, simply fails to even suggest "receiving additional instructions... in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 64, 70 and 74-75, the Examiner has argued the following:

"Rivard also teaches a texture mapping stage as shown in Fig. 6 receiving data (instructions) from the rasterizer module of the pipeline and then receiving data (instructions) from the DRAM (it should be noted that the data has it's associated instructions; texture mapping stage in the graphics pipeline receives data from the rasterizer module to perform texturing/rendering operation; this data received from the rasterizer module is considered to be primary data). Therefore, the data received from the DRAM via the texel cache system is considered to be the additional data (Wang reference as shown below teaches that the instructions include data)."

Appellant respectfully disagrees and asserts that Rivard teaches that "[g]raphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650" and that "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 49-57 – emphasis added). Further, Wang teaches that "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" (Col. 5, lines 40-43 – emphasis added).

However, texture mapping stage 645 mapping **texture information** to the graphics information received from graphics application program 670 and maintaining **the texel information in a texel cache system 650**, where texture mapping block 645 sends information to texel cache system 650, and texel cache system 650 sends information back to texture mapping block 645, as in Rivard, in addition to a 3D graphics subunit 109 executing a series of display instructions found within a display list stored in computer memory, as in Wang, simply fails to suggest "receiving additional instructions... in response to the instruction request utilizing the texture module" (emphasis added), as claimed by appellant. Clearly, a texture mapping stage 645 sending information to texel cache system 650, and the texel cache system 650 sending information back to texture mapping block 645, as in Rivard, simply fails to suggest "receiving additional instructions... in response to the instruction request" (emphasis added), as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 64-65, 70-71 and 75, the Examiner has argued the following:

"Rivard also teaches the arrival of memory data and associated instructions at the cache data store and memory data resolver component of the text cache system is coordinated by pipeline latency elements (col. 7 lines 4-7; memory data has associated instructions; it should be noted that depending on the DRAM design, there could be a constant latency to effect a page hit; however, in the instance when the instructions and data are combined together, no pipeline latency element will be required; therefore, when the Rivard reference is modified as taught by Wang below, the modified system might have the data and the instructions coordinated when arrived at cache data store and memory data resolver, and will not require the use of pipeline latency element; however, having the pipeline latency element will help to coordinate the instructions and data at their arrival at cache data store and memory data resolver, when there is a delay between the arrival of data and it's associated instructions; it should be noted that Rivard has this pipeline latency element in addition to the claimed limitation, and moreover, the examiner interprets that the modified system with pipeline latency element will still function as effectively and generate the desired results, even when there is no delay between the arrival of the data and it's associated instructions)."

Appellant respectfully disagrees and asserts that Wang merely teaches that "[t]he graphics hardware system 108, over bus 100", supplies data and control signals to a local frame buffer memory 110" (Col. 5, lines 63-67) and that the "texture map data access (TDA) circuit 200... provides an efficient mechanism for processing texture map data requests (in the form of addresses) whereby useful texture map data can be supplied from a cache memory 251 to the filter 260 simultaneously during a fetch interval wherein other texture data is being fetched from main memory 102 or from a local frame buffer 110 (FIG. 1)" (Col. 6, lines 53-61 – emphasis added). Further, Wang teaches that "FIFO memory 240 of the TDA circuit 200" (Col. 8, lines 38-39), "...contains a number of entries, (1)-(m), for storing hit or miss addresses" and that "[i]f the texture map data corresponding to the bottom entry address is not yet available in the cache memory 251, FIFO memory 240 is stalled until the data becomes available" (Col. 8, lines 14-16 and 22-25 – emphasis added). Therefore, Wang teaches that the texture map data access circuit processes texture map data requests and supplies texture map data from a cache memory 251, where a FIFO memory 240 of the texture map data access circuit, which stores hit or miss addresses, stalls if the texture map data corresponding to the bottom entry address is not yet available in the cache memory 251.

Furthermore, Rivard teaches that "[c]ache tag blocks 1010 and 1015 each determine whether requested texel values are stored in cache data store 1030," where "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" that

"generates memory requests for all misses... and <u>forwards the requests to DRAM 655</u> for <u>information retrieval</u>" and that the "pipeline latency elements 1025... coordinate <u>arrival of the memory data</u> and of the <u>associated instructions</u> at cache data store. and memory data resolver 1030" (Col. 6, line 40-Col. 7, line 7 – emphasis added). Therefore, Rivard's <u>pipeline latency elements 1025</u> coordinate arrival of <u>the requested texel values</u> <u>retrieved from DRAM</u> and the arrival of the <u>associated instructions</u> at cache data store 1030.

However, Wang's texture map data access circuit that <u>processes texture map data requests</u> and supplies <u>texture map data</u> <u>from a cache memory 251</u>, where a FIFO memory 240 of the texture map data access circuit stalls if the **texture map data** corresponding to the bottom entry address <u>is not yet available in the cache memory 251</u>, in addition to Rivard's <u>pipeline latency elements 1025</u> that coordinate arrival of <u>the requested texel values</u> <u>retrieved from DRAM</u> and the arrival of the <u>associated instructions</u> at cache data store 1030, fails to support the Examiner's allegation that the "when the Rivard reference is modified as taught by Wang below, the modified system <u>might have the data and the instructions coordinated when arrived at cache data store and memory data resolver</u>, and will not require the use of pipeline latency element" (emphasis added), as alleged by the Examiner.

Further, a <u>texture map data access circuit</u> that processes <u>texture map data requests</u> and supplies <u>texture map data</u> <u>from a cache memory 251</u>, as in Wang, in addition <u>a memory request generator 1020</u> that generates memory requests for all misses and forwards the requests to DRAM 655 for information retrieval, where pipeline latency elements 1025 to coordinating arrival of <u>the requested texel values</u> <u>retrieved from DRAM</u> and the arrival of the <u>associated instructions</u> at cache data store 1030, as in Rivard, simply fails to suggest "receiving <u>additional instructions</u>... in response to the instruction request <u>utilizing the texture module</u>" (emphasis added), as claimed by appellant. Clearly, a <u>texture map data access circuit</u> processing <u>texture map data requests</u>, as in Wang, in addition to a <u>memory request generator</u> generating <u>memory requests for texel values</u> for all misses and forwarding the requests to DRAM <u>for texel value retrieval</u>, as in Rivard, simply fails to suggest that "receiving <u>additional instructions</u>... in response to the instruction request <u>utilizing the texture module</u>" (emphasis added), as claimed by appellant.

In the Examiner's Answer mailed 06/26/2008, in the paragraph on Pages 65-66, 71-72 and 76-77, the Examiner has argued the following:

> "However, Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM. Therefore, the examiner brings in the Wang reference that suggests a graphics subunit (texture module) in a graphics hardware system that supplies data and control signals to local frame buffer memory for executing a series of display instructions (Fig. 1, col. 5 lines 38-67; the computer memory includes the local frame buffer memory, which corresponds to the DRAM; based on the control signals send by the graphics hardware system, the frame buffer returns the polygon display instructions that includes the texture data, so that the graphics subunit executes the display instructions using this data; it should be noted that Wang teaches to return display instructions from the display list to the graphics hardware system for further processing, where the display instructions include texture data; it should be noted that the instructions and data returned by the display list to the graphics hardware system is considered additional instructions or data as the Rivard reference already showed that a rasterizer module passes on preliminary instructions or data to the texture module; it should also be noted that the graphics hardware system includes a 3D graphics subunit that includes a texture engine that is responsible for retrieving the texture map data for the polygon and mapping the texels of the texture data onto the pixels of the polygon; this texture engine of Wang is functional equivalent of the texture module as suggested by Rivard). Therefore, it would have been obvious to one of ordinary skill in art at the time of present invention to have the memory return instructions as taught by Wang to the texture module of Rivard because these instructions are needed to render the graphics primitives to be displayed on the display device (col. 5 lines 43-45 and lines 63-67)."

Appellant agrees with the Examiner's statement that "Rivard does not explicitly state that the data and it's associated instructions are returned by DRAM," as stated by the Examiner. Further, appellant respectfully disagrees with the Examiner's assertions made with respect to the Wang reference.

Specifically, appellant asserts that Wang merely teaches that "the graphics subunit 109 includ[es] a texture engine 10, a polygon engine 12 and a pixel pipeline 16" (Col. 6, lines 3-5 – emphasis added) and that Fig. 2 illustrates that the texture engine 10 further contains a texture map data access circuit 200, where the "[t]exture map data retrieval (TDA) circuit 200 performs texture map data retrieval processes" (Col. 6, lines 23-25 – emphasis added). Further, Wang teaches that the "[t]exture data is stored in computer readable (e.g., volatile) memory units of system 112, or local frame buffer 110" (Col. 5, lines 38-51 – emphasis added). Additionally, Wang teaches that "[t]he host computer system 112 provides data and control signals via bus 100 to a graphics hardware unit or system, e.g., 'graphics card' 108," where "[t]he graphics hardware system 108

contains <u>a 3D graphics subunit 109 for executing a series of display instructions</u> found within <u>a</u> <u>display list stored in computer memory</u>" (Col. 5, lines 38-43 – emphasis added). In addition, Wang teaches that "[t]he [display] instructions <u>define the rendering of several graphic primitives</u>, e.g., individual points, lines, polygons, fills, BIT BLTs (bit block transfers), <u>textures</u>, etc., and graphics commands" (Col. 5, lines 43-46 – emphasis added).

However, <u>the host computer system providing data</u> and control signals via bus 100 <u>to a graphics</u> <u>hardware unit</u> that contains a 3D graphics subunit 109 for <u>executing a series of display</u> <u>instructions</u>, which <u>define the rendering of several graphic primitives</u>, found within a display list stored in computer memory, in addition to <u>a texture map data retrieval (TDA) circuit</u> of the texture engine performing <u>texture map data retrieval processes</u>, where <u>texture data is stored</u> in memory units of the system 112 or <u>the local frame buffer 110</u>, as in Wang, fails to support the Examiner's assertion that Wang suggests "a graphics subunit (<u>texture module</u>) in a graphics hardware system that <u>supplies data and control signals to local frame buffer memory for</u> <u>executing a series of display instructions</u>," as asserted by the Examiner, and further fails to even suggest "receiving <u>additional instructions</u>... <u>in response to the instruction request</u> utilizing the texture module" (emphasis added), as claimed by appellant.

Furthermore, a 3D graphics subunit 109 for <u>executing a series of display instructions</u> that <u>define</u> <u>the rendering of several graphic primitives</u>, such as textures, in addition to performing the retrieval process of <u>texture map data</u> <u>stored</u> in <u>the local frame buffer 110</u>, as in Wang, simply fails to suggest "receiving <u>additional instructions</u>... <u>in response to the instruction request</u> utilizing the texture module" (emphasis added), as claimed by appellant. Clearly, executing a series of display instructions that <u>define the rendering of graphic primitives such as textures</u>, in addition to **<u>retrieving texture map data</u>** stored in a local frame buffer, as in Wang, fails to suggest "receiving <u>additional instructions</u>" (emphasis added), as claimed by appellant, since Wang's retrieval process merely retrieves <u>texture map data</u>.

Again, appellant respectfully asserts that at least the first and third elements of the *prima facie* case of obviousness have not been met, since it would be *unobvious* to combine the references, as noted above, and the prior art excerpts, as relied upon by the Examiner, fail to teach or suggest <u>all</u> of the claim limitations, as noted above.

In view of the remarks set forth hereinabove, all of the independent claims are deemed allowable, along with any claims depending therefrom.

In the event a telephone conversation would expedite the prosecution of this application, the Examiner may reach the undersigned at (408) 971-2573. For payment of any additional fees due in connection with the filing of this paper, the Commissioner is authorized to charge such fees to Deposit Account No. 50-1351 (Order No. NVIDP064).

Respectfully submitted,

By: __/KEVINZILKA/_____   Date: __August 26, 2008_____
Kevin J. Zilka
Reg. No. 41,429

Zilka-Kotab, P.C.
P.O. Box 721120
San Jose, California 95172-1120
Telephone: (408) 971-2573
Facsimile: (408) 971-4660